REPUBLIQUE DU CAMEROUN

Paix - Travail - Patrie

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR

REPUBLIC OF CAMEROON

Peace - Work - Fatherland

MINISTRY OF HIGHER EDUCATION

CORRIGE - BTS ECM Session 2026

PRATIQUE DE PROGRAMMATION - SUJET 2

Cas : FOODFAST CM - Module Avis et Notations

PARTIE A : ECRITE (10 points)

SECTION 1 : ALGORITHMIQUE (5 points)

Exercice 1 : Calcul des frais de livraison (2,5 points)

1.1. Algorithme CalculerFraisLivraison (1,5 pt)

```
ALGORITHME CalculerFraisLivraison
VARIABLES
   distance : REEL
   montant_commande : REEL
   frais : ENTIER
DEBUT
   LIRE(distance, montant_commande)

   // Livraison gratuite si commande > 15000 FCFA
   SI montant_commande > 15000 ALORS
      frais <- 0
   SINON
      // Calcul selon la distance
      SI distance <= 2 ALORS
         frais <- 500
      SINON SI distance <= 5 ALORS
         frais <- 800
      SINON SI distance <= 10 ALORS
         frais <- 1200
      SINON
         frais <- 1500
      FIN SI
   FIN SI

   RETOURNER frais
FIN
```

1.2. Traces d'execution (1 pt)

Test 1 : distance = 7, montant = 12000

```
Etape 1: distance = 7, montant_commande = 12000
Etape 2: 12000 <= 15000, donc on calcule les frais
Etape 3: 7 > 5 et 7 <= 10, donc frais = 1200
Resultat: RETOURNER 1200 FCFA
```

Test 2 : distance = 3, montant = 18000

```
Etape 1: distance = 3, montant_commande = 18000
Etape 2: 18000 > 15000, donc livraison gratuite
Etape 3: frais = 0
Resultat: RETOURNER 0 FCFA (livraison gratuite)
```

Exercice 2 : Calcul de la moyenne des notes (2,5 points)

2.1. Algorithme CalculerMoyenneNotes (1,5 pt)

```
ALGORITHME CalculerMoyenneNotes
VARIABLES
   notes : TABLEAU D'ENTIERS
   i, somme, compteur : ENTIER
   moyenne : REEL
DEBUT
   somme <- 0
   compteur <- 0

   POUR i DE 1 A TAILLE(notes) FAIRE
      // Ignorer les notes invalides
      SI notes[i] >= 1 ET notes[i] <= 5 ALORS
         somme <- somme + notes[i]
         compteur <- compteur + 1
      FIN SI
   FIN POUR

   SI compteur = 0 ALORS
      moyenne <- 0
   SINON
      moyenne <- ARRONDI(somme / compteur, 1)
   FIN SI

   RETOURNER moyenne
FIN
```

2.2. Algorithme CompterEtoiles (1 pt)

```
ALGORITHME CompterEtoiles
VARIABLES
   notes : TABLEAU D'ENTIERS
   distribution : TABLEAU[1..5] D'ENTIERS
   i : ENTIER
DEBUT
   // Initialiser le tableau de distribution
   POUR i DE 1 A 5 FAIRE
      distribution[i] <- 0
   FIN POUR

   // Compter les occurrences
   POUR i DE 1 A TAILLE(notes) FAIRE
      SI notes[i] >= 1 ET notes[i] <= 5 ALORS
         distribution[notes[i]] <- distribution[notes[i]] + 1
      FIN SI
   FIN POUR

   // Afficher la distribution
   POUR i DE 5 A 1 PAS -1 FAIRE
      AFFICHER(i, " etoile(s): ", distribution[i], " avis")
   FIN POUR
FIN
```

SECTION 2 : ANALYSE ET CONCEPTION (5 points)

Exercice 3 : MCD et MLD (2,5 points)

3.1. Schema MCD (2 pts)

Entites et attributs :

```
CLIENT (id_client, nom, prenom, email, telephone, adresse)
COMMANDE (id_commande, date_commande, montant_total, frais_livraison, statut)
RESTAURANT (id_restaurant, nom, adresse, telephone, categorie_cuisine, note_moyenne)
LIVREUR (id_livreur, nom, prenom, telephone, vehicule, note_moyenne, disponible)
PLAT (id_plat, nom, description, prix, disponible)
CATEGORIE (id_categorie, nom) -- entree, plat, dessert, boisson
AVIS (id_avis, note_restaurant, note_livreur, commentaire, date_avis, signale)
```

Associations et cardinalites :

```
CLIENT ----(1,n)---- PASSER ----(1,1)---- COMMANDE
RESTAURANT ----(1,n)---- RECEVOIR ----(1,1)---- COMMANDE
LIVREUR ----(0,n)---- LIVRER ----(1,1)---- COMMANDE
COMMANDE ----(1,n)---- CONTENIR ----(0,n)---- PLAT [+ quantite]
PLAT ----(1,1)---- APPARTENIR ----(1,n)---- CATEGORIE
PLAT ----(1,1)---- PROPOSER ----(1,n)---- RESTAURANT
COMMANDE ----(0,1)---- AVOIR ----(1,1)---- AVIS
```

3.2. MLD - Schema relationnel (0,5 pt)

```
CLIENT (id_client PK, nom, prenom, email, telephone, adresse)
RESTAURANT (id_restaurant PK, nom, adresse, telephone, categorie, note_moyenne)
LIVREUR (id_livreur PK, nom, prenom, telephone, vehicule, note_moyenne)
CATEGORIE (id_categorie PK, nom)
PLAT (id_plat PK, nom, description, prix, #id_restaurant FK, #id_categorie FK)
COMMANDE (id_commande PK, date, montant, frais, statut,
          #id_client FK, #id_restaurant FK, #id_livreur FK)
LIGNE_COMMANDE (id_ligne PK, quantite, #id_commande FK, #id_plat FK)
AVIS (id_avis PK, note_resto, note_livreur, commentaire, date, #id_commande FK UNIQUE)
```

Exercice 4 : Diagramme de sequence (2,5 points)

4.1. Acteurs et objets (0,5 pt)

```
Acteurs : Client
Objets : ApplicationMobile, ServeurAPI, BaseDeDonnees, ServiceNotification
```

4.2. Description du diagramme de sequence (2 pts)

```
1. ServiceNotification -> Client : envoyerNotification("Donnez votre avis")
2. Client -> ApplicationMobile : ouvrirApplication()
3. ApplicationMobile -> ServeurAPI : getCommandeLivree(id_client)
4. ServeurAPI -> BaseDeDonnees : SELECT commande WHERE statut='livree'
5. BaseDeDonnees --> ServeurAPI : donneesCommande
6. ServeurAPI --> ApplicationMobile : afficherDetailsCommande()
7. Client -> ApplicationMobile : selectionnerNoteRestaurant(4)
8. Client -> ApplicationMobile : selectionnerNoteLivreur(5)
9. Client -> ApplicationMobile : saisirCommentaire("Tres bon service")
10. Client -> ApplicationMobile : validerAvis()
11. ApplicationMobile -> ServeurAPI : POST /avis {notes, commentaire}
12. ServeurAPI -> BaseDeDonnees : INSERT INTO avis...
13. ServeurAPI -> BaseDeDonnees : UPDATE restaurant SET note_moyenne...
14. ServeurAPI -> BaseDeDonnees : UPDATE livreur SET note_moyenne...
15. BaseDeDonnees --> ServeurAPI : confirmation
16. ServeurAPI --> ApplicationMobile : {success: true}
17. ApplicationMobile --> Client : afficherConfirmation("Merci !")
```

PARTIE B : PRATIQUE (40 points)

MODULE 1 : BASE DE DONNEES MySQL (10 points)

Exercice 1 : Creation de la base (4 points)

1.1. Creation de la base (0,5 pt)

```
CREATE DATABASE foodfast_avis;
USE foodfast_avis;
```

1.2. Creation des tables (3,5 pts)

```
CREATE TABLE restaurants (
   id_restaurant INT AUTO_INCREMENT PRIMARY KEY,
   nom VARCHAR(100) NOT NULL,
   adresse VARCHAR(255) NOT NULL,
   telephone VARCHAR(20),
   categorie_cuisine VARCHAR(50),
   note_moyenne DECIMAL(2,1) DEFAULT 0.0,
   nb_avis INT DEFAULT 0,
   actif BOOLEAN DEFAULT TRUE,
   CHECK (note_moyenne >= 0 AND note_moyenne <= 5)
);

CREATE TABLE livreurs (
   id_livreur INT AUTO_INCREMENT PRIMARY KEY,
   nom VARCHAR(50) NOT NULL,
   prenom VARCHAR(50) NOT NULL,
   telephone VARCHAR(20) NOT NULL,
   vehicule ENUM('moto', 'velo', 'voiture') NOT NULL,
   note_moyenne DECIMAL(2,1) DEFAULT 0.0,
   nb_livraisons INT DEFAULT 0,
   disponible BOOLEAN DEFAULT TRUE
);

CREATE TABLE commandes (
   id_commande INT AUTO_INCREMENT PRIMARY KEY,
   id_client INT NOT NULL,
   id_restaurant INT NOT NULL,
   id_livreur INT,
   date_commande DATETIME DEFAULT CURRENT_TIMESTAMP,
   montant_total DECIMAL(10,2) NOT NULL,
   frais_livraison DECIMAL(10,2) DEFAULT 0,
   statut ENUM('en_attente', 'en_preparation', 'en_livraison',
               'livree', 'annulee') DEFAULT 'en_attente',
   FOREIGN KEY (id_restaurant) REFERENCES restaurants(id_restaurant),
   FOREIGN KEY (id_livreur) REFERENCES livreurs(id_livreur)
);

CREATE TABLE avis (
   id_avis INT AUTO_INCREMENT PRIMARY KEY,
   id_commande INT NOT NULL UNIQUE,
   note_restaurant TINYINT NOT NULL CHECK (note_restaurant BETWEEN 1 AND 5),
   note_livreur TINYINT NOT NULL CHECK (note_livreur BETWEEN 1 AND 5),
   commentaire TEXT,
   date_avis DATETIME DEFAULT CURRENT_TIMESTAMP,
   signale BOOLEAN DEFAULT FALSE,
   FOREIGN KEY (id_commande) REFERENCES commandes(id_commande)
);
```

Exercice 2 : Requetes SQL (6 points)

2.1. Insertion des donnees de test (1 pt)

```
INSERT INTO restaurants (nom, adresse, telephone, categorie_cuisine) VALUES
('Le Maquis du Coin', 'Bastos, Yaounde', '677123456', 'camerounaise'),
('Pizza Express', 'Centre-ville, Yaounde', '699888777', 'italienne'),
('Grill Master', 'Mvan, Yaounde', '655444333', 'grillades'),
('Chez Mama', 'Mvog-Mbi, Yaounde', '677999888', 'camerounaise'),
('Burger House', 'Omnisport, Yaounde', '699111222', 'fast-food');

INSERT INTO livreurs (nom, prenom, telephone, vehicule) VALUES
('Mbarga', 'Jean', '677111222', 'moto'),
('Nkolo', 'Pierre', '699333444', 'velo'),
('Ateba', 'Paul', '655666777', 'moto');
```

2.2. TOP 5 des restaurants (1 pt)

```
SELECT nom, categorie_cuisine, note_moyenne, nb_avis
FROM restaurants
WHERE nb_avis >= 10 AND actif = TRUE
ORDER BY note_moyenne DESC, nb_avis DESC
LIMIT 5;
```

2.3. Note moyenne par categorie (1,5 pt)

```
SELECT r.categorie_cuisine,
       ROUND(AVG(a.note_restaurant), 1) AS moyenne_notes,
       COUNT(a.id_avis) AS nombre_avis
FROM restaurants r
INNER JOIN commandes c ON r.id_restaurant = c.id_restaurant
INNER JOIN avis a ON c.id_commande = a.id_commande
GROUP BY r.categorie_cuisine
ORDER BY moyenne_notes DESC;
```

2.4. TRIGGER mise a jour automatique (2,5 pts)

```
DELIMITER //
CREATE TRIGGER after_avis_insert
AFTER INSERT ON avis
FOR EACH ROW
BEGIN
   DECLARE v_id_restaurant INT;
   DECLARE v_id_livreur INT;
   DECLARE v_nouvelle_moyenne_resto DECIMAL(2,1);
   DECLARE v_nouvelle_moyenne_livreur DECIMAL(2,1);

   -- Recuperer les IDs depuis la commande
   SELECT id_restaurant, id_livreur INTO v_id_restaurant, v_id_livreur
   FROM commandes WHERE id_commande = NEW.id_commande;

   -- Calculer nouvelle moyenne restaurant
   SELECT ROUND(AVG(a.note_restaurant), 1)
   INTO v_nouvelle_moyenne_resto
   FROM avis a
   INNER JOIN commandes c ON a.id_commande = c.id_commande
   WHERE c.id_restaurant = v_id_restaurant;

   -- Mettre a jour le restaurant
   UPDATE restaurants
   SET note_moyenne = v_nouvelle_moyenne_resto,
       nb_avis = nb_avis + 1
   WHERE id_restaurant = v_id_restaurant;

   -- Calculer et mettre a jour moyenne livreur si existe
```

```
   IF v_id_livreur IS NOT NULL THEN
      SELECT ROUND(AVG(a.note_livreur), 1)
      INTO v_nouvelle_moyenne_livreur
      FROM avis a
      INNER JOIN commandes c ON a.id_commande = c.id_commande
      WHERE c.id_livreur = v_id_livreur;

      UPDATE livreurs
      SET note_moyenne = v_nouvelle_moyenne_livreur
      WHERE id_livreur = v_id_livreur;
   END IF;
END //
DELIMITER ;
```

MODULE 2 : DEVELOPPEMENT BACKEND PHP (15 points)

Exercice 3 : Classe Avis.php (8 points)

```php
<?php
class Avis {
   private $pdo;
   private $id_avis;
   private $id_commande;
   private $note_restaurant;
   private $note_livreur;
   private $commentaire;
   private $date_avis;
   private $signale;

   public function __construct() {
      try {
         $this->pdo = new PDO(
            'mysql:host=localhost;dbname=foodfast_avis;charset=utf8',
            'root', '',
            [PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION]
         );
      } catch (PDOException $e) {
         die('Erreur connexion: ' . $e->getMessage());
      }
   }

   // Getters et Setters
   public function getNoteRestaurant() { return $this->note_restaurant; }
   public function setNoteRestaurant($note) {
      if ($this->validerNote($note)) $this->note_restaurant = $note;
   }
   public function getNoteLivreur() { return $this->note_livreur; }
   public function setNoteLivreur($note) {
      if ($this->validerNote($note)) $this->note_livreur = $note;
   }

   // Valider une note (entre 1 et 5)
   public function validerNote($note) {
      return is_numeric($note) && $note >= 1 && $note <= 5;
   }

   // Deposer un avis
   public function deposerAvis($id_commande, $note_resto, $note_livreur, $commentaire = '') {
      // Validation
      if (!$this->validerNote($note_resto) || !$this->validerNote($note_livreur)) {
         return ['success' => false, 'message' => 'Notes invalides (1-5)'];
      }

      // Verifier que la commande existe et n'a pas deja d'avis
      $check = $this->pdo->prepare("SELECT id_commande FROM commandes
                WHERE id_commande = ? AND statut = 'livree'");
      $check->execute([$id_commande]);
      if (!$check->fetch()) {
         return ['success' => false, 'message' => 'Commande invalide ou non livree'];
      }

      // Verifier si avis existe deja
      $checkAvis = $this->pdo->prepare("SELECT id_avis FROM avis WHERE id_commande = ?");
      $checkAvis->execute([$id_commande]);
      if ($checkAvis->fetch()) {
         return ['success' => false, 'message' => 'Avis deja depose'];
      }
```

```
        // Inserer l'avis
        $sql = "INSERT INTO avis (id_commande, note_restaurant, note_livreur, commentaire)
                VALUES (?, ?, ?, ?)";
        $stmt = $this->pdo->prepare($sql);

        if ($stmt->execute([$id_commande, $note_resto, $note_livreur, $commentaire])) {
            return ['success' => true, 'message' => 'Avis enregistre', 'id' =>
$this->pdo->lastInsertId()];
        }
        return ['success' => false, 'message' => 'Erreur enregistrement'];
    }

    // Recuperer les avis d'un restaurant
    public function getAvisRestaurant($id_restaurant, $limit = 10) {
        $sql = "SELECT a.*, c.date_commande
                FROM avis a
                INNER JOIN commandes c ON a.id_commande = c.id_commande
                WHERE c.id_restaurant = ? AND a.signale = FALSE
                ORDER BY a.date_avis DESC
                LIMIT ?";
        $stmt = $this->pdo->prepare($sql);
        $stmt->execute([$id_restaurant, $limit]);
        return $stmt->fetchAll(PDO::FETCH_ASSOC);
    }

    // Calculer la moyenne d'un restaurant
    public function getMoyenneRestaurant($id_restaurant) {
        $sql = "SELECT ROUND(AVG(a.note_restaurant), 1) as moyenne,
                       COUNT(*) as nb_avis
                FROM avis a
                INNER JOIN commandes c ON a.id_commande = c.id_commande
                WHERE c.id_restaurant = ?";
        $stmt = $this->pdo->prepare($sql);
        $stmt->execute([$id_restaurant]);
        return $stmt->fetch(PDO::FETCH_ASSOC);
    }

    // Signaler un avis
    public function signalerAvis($id_avis) {
        $sql = "UPDATE avis SET signale = TRUE WHERE id_avis = ?";
        $stmt = $this->pdo->prepare($sql);
        return $stmt->execute([$id_avis]);
    }

    // Distribution des notes
    public function getDistribution($id_restaurant) {
        $sql = "SELECT a.note_restaurant as note, COUNT(*) as count
                FROM avis a
                INNER JOIN commandes c ON a.id_commande = c.id_commande
                WHERE c.id_restaurant = ?
                GROUP BY a.note_restaurant
                ORDER BY note DESC";
        $stmt = $this->pdo->prepare($sql);
        $stmt->execute([$id_restaurant]);
        return $stmt->fetchAll(PDO::FETCH_ASSOC);
    }
}
?>
```

Exercice 4 : API REST api_avis.php (7 points)

```php
<?php
header('Content-Type: application/json; charset=utf-8');
header('Access-Control-Allow-Origin: *');
header('Access-Control-Allow-Methods: GET, POST, OPTIONS');

require_once 'Avis.php';

$avis = new Avis();
$action = $_GET['action'] ?? '';
$method = $_SERVER['REQUEST_METHOD'];

try {
   switch ($action) {

      case 'restaurant':
         // GET /api_avis.php?action=restaurant&id=X
         if ($method !== 'GET') throw new Exception('Methode non autorisee', 405);
         $id = intval($_GET['id'] ?? 0);
         if ($id <= 0) throw new Exception('ID restaurant requis', 400);
         $limit = intval($_GET['limit'] ?? 10);
         $data = $avis->getAvisRestaurant($id, $limit);
         echo json_encode(['success' => true, 'data' => $data]);
         break;

      case 'stats':
         // GET /api_avis.php?action=stats&id=X
         if ($method !== 'GET') throw new Exception('Methode non autorisee', 405);
         $id = intval($_GET['id'] ?? 0);
         if ($id <= 0) throw new Exception('ID restaurant requis', 400);
         $moyenne = $avis->getMoyenneRestaurant($id);
         $distribution = $avis->getDistribution($id);
         echo json_encode([
            'success' => true,
            'moyenne' => $moyenne['moyenne'],
            'nb_avis' => $moyenne['nb_avis'],
            'distribution' => $distribution
         ]);
         break;

      case 'deposer':
         // POST /api_avis.php?action=deposer
         if ($method !== 'POST') throw new Exception('Methode non autorisee', 405);
         $data = json_decode(file_get_contents('php://input'), true);
         $id_commande = intval($data['id_commande'] ?? 0);
         $note_resto = intval($data['note_restaurant'] ?? 0);
         $note_livreur = intval($data['note_livreur'] ?? 0);
         $commentaire = trim($data['commentaire'] ?? '');

         if ($id_commande <= 0 || $note_resto <= 0 || $note_livreur <= 0) {
            throw new Exception('Donnees invalides', 400);
         }

         $result = $avis->deposerAvis($id_commande, $note_resto, $note_livreur,
$commentaire);
         if (!$result['success']) {
            throw new Exception($result['message'], 400);
         }
         echo json_encode($result);
         break;
```

```
      case 'signaler':
         // POST /api_avis.php?action=signaler
         if ($method !== 'POST') throw new Exception('Methode non autorisee', 405);
         $data = json_decode(file_get_contents('php://input'), true);
         $id_avis = intval($data['id_avis'] ?? 0);
         if ($id_avis <= 0) throw new Exception('ID avis requis', 400);

         if ($avis->signalerAvis($id_avis)) {
            echo json_encode(['success' => true, 'message' => 'Avis signale']);
         } else {
            throw new Exception('Erreur signalement', 500);
         }
         break;

      default:
         throw new Exception('Action non reconnue', 404);
   }
} catch (Exception $e) {
   $code = $e->getCode() ?: 500;
   http_response_code($code);
   echo json_encode(['success' => false, 'error' => $e->getMessage()]);
}
?>
```

MODULE 3 : DEVELOPPEMENT FRONTEND (15 points)

Exercice 5 : Page de depot d'avis (8 points)

5.1. deposer_avis.html (3 pts)

```
<!DOCTYPE html>
<html lang="fr">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Donner mon avis - FoodFast</title>
   <link rel="stylesheet" href="style_avis.css">
</head>
<body>
   <div class="container">
      <h1>Donnez votre avis</h1>

      <div class="commande-resume">
         <h2>Votre commande</h2>
         <p><strong>Restaurant:</strong> <span id="nomRestaurant">Le Maquis</span></p>
         <p><strong>Livreur:</strong> <span id="nomLivreur">Jean M.</span></p>
         <p><strong>Date:</strong> <span id="dateCommande">15/01/2024</span></p>
      </div>

      <form id="formAvis">
         <input type="hidden" id="idCommande" value="123">

         <div class="rating-section">
            <h3>Note du restaurant</h3>
            <div class="stars" id="starsRestaurant" data-rating="0">
               <span class="star" data-value="1">&#9733;</span>
               <span class="star" data-value="2">&#9733;</span>
               <span class="star" data-value="3">&#9733;</span>
               <span class="star" data-value="4">&#9733;</span>
               <span class="star" data-value="5">&#9733;</span>
            </div>
            <input type="hidden" id="noteRestaurant" name="note_restaurant" value="0">
         </div>

         <div class="rating-section">
            <h3>Note du livreur</h3>
            <div class="stars" id="starsLivreur" data-rating="0">
               <span class="star" data-value="1">&#9733;</span>
               <span class="star" data-value="2">&#9733;</span>
               <span class="star" data-value="3">&#9733;</span>
               <span class="star" data-value="4">&#9733;</span>
               <span class="star" data-value="5">&#9733;</span>
            </div>
            <input type="hidden" id="noteLivreur" name="note_livreur" value="0">
         </div>

         <div class="comment-section">
            <h3>Commentaire (optionnel)</h3>
            <textarea id="commentaire" maxlength="500" placeholder="Partagez votre
experience..."></textarea>
            <div class="char-count"><span id="charCount">0</span>/500</div>
         </div>

         <button type="submit" id="btnEnvoyer">Envoyer mon avis</button>
      </form>

      <div id="messageResult" class="message"></div>
```

```
   </div>
   <script src="avis.js"></script>
</body>
</html>
```

5.2. style_avis.css (2,5 pts)

```
* { margin: 0; padding: 0; box-sizing: border-box; }
body { font-family: Arial; background: #f8f8f8; min-height: 100vh; }
.container { max-width: 500px; margin: 20px auto; padding: 20px;
             background: white; border-radius: 15px; box-shadow: 0 4px 15px
rgba(0,0,0,0.1); }
h1 { color: #E74C3C; text-align: center; margin-bottom: 20px; }
.commande-resume { background: #fef9f9; padding: 15px; border-radius: 10px;
                   border-left: 4px solid #E74C3C; margin-bottom: 25px; }
.rating-section { margin-bottom: 25px; }
.rating-section h3 { color: #333; margin-bottom: 10px; }
.stars { display: flex; gap: 5px; }
.star { font-size: 40px; color: #ddd; cursor: pointer; transition: all 0.2s; }
.star:hover, .star.active { color: #FFD700; transform: scale(1.2); }
.star.hover { color: #FFE066; }
.comment-section textarea { width: 100%; height: 100px; padding: 12px;
                            border: 2px solid #ddd; border-radius: 8px;
                            resize: none; font-size: 14px; }
.comment-section textarea:focus { border-color: #E74C3C; outline: none; }
.char-count { text-align: right; color: #999; font-size: 12px; margin-top: 5px; }
#btnEnvoyer { width: 100%; padding: 15px; background: #E74C3C; color: white;
              font-size: 18px; border: none; border-radius: 10px; cursor: pointer;
              transition: background 0.3s, transform 0.2s; }
#btnEnvoyer:hover { background: #C0392B; transform: translateY(-2px); }
#btnEnvoyer:disabled { background: #ccc; cursor: not-allowed; }
.message { margin-top: 15px; padding: 15px; border-radius: 8px; text-align: center; }
.message.success { background: #d4edda; color: #155724; }
.message.error { background: #f8d7da; color: #721c24; }
@keyframes fadeIn { from { opacity: 0; transform: translateY(-10px); }
                    to { opacity: 1; transform: translateY(0); } }
.message { animation: fadeIn 0.3s ease; }
@media (max-width: 480px) { .container { margin: 10px; padding: 15px; }
   .star { font-size: 32px; } }
```

5.3. avis.js (2,5 pts)

```
document.addEventListener('DOMContentLoaded', function() {
   const starsResto = document.querySelectorAll('#starsRestaurant .star');
   const starsLivreur = document.querySelectorAll('#starsLivreur .star');
   const inputNoteResto = document.getElementById('noteRestaurant');
   const inputNoteLivreur = document.getElementById('noteLivreur');
   const textarea = document.getElementById('commentaire');
   const charCount = document.getElementById('charCount');
   const form = document.getElementById('formAvis');
   const messageResult = document.getElementById('messageResult');

   // Gestion des etoiles
   function setupStars(stars, input) {
      stars.forEach(star => {
         star.addEventListener('click', function() {
            const value = this.dataset.value;
            input.value = value;
            stars.forEach(s => {
               s.classList.toggle('active', s.dataset.value <= value);
            });
         });
         star.addEventListener('mouseenter', function() {
```

```
            const value = this.dataset.value;
            stars.forEach(s => {
               s.classList.toggle('hover', s.dataset.value <= value);
            });
         });
         star.addEventListener('mouseleave', function() {
            stars.forEach(s => s.classList.remove('hover'));
         });
      });
   }

   setupStars(starsResto, inputNoteResto);
   setupStars(starsLivreur, inputNoteLivreur);

   // Compteur de caracteres
   textarea.addEventListener('input', function() {
      charCount.textContent = this.value.length;
   });

   // Soumission du formulaire
   form.addEventListener('submit', async function(e) {
      e.preventDefault();

      const noteResto = parseInt(inputNoteResto.value);
      const noteLivreur = parseInt(inputNoteLivreur.value);

      // Validation
      if (noteResto < 1 || noteResto > 5) {
         showMessage('Veuillez noter le restaurant', 'error'); return;
      }
      if (noteLivreur < 1 || noteLivreur > 5) {
         showMessage('Veuillez noter le livreur', 'error'); return;
      }

      // Envoi API
      try {
         const response = await fetch('api_avis.php?action=deposer', {
            method: 'POST',
            headers: { 'Content-Type': 'application/json' },
            body: JSON.stringify({
               id_commande: document.getElementById('idCommande').value,
               note_restaurant: noteResto,
               note_livreur: noteLivreur,
               commentaire: textarea.value
            })
         });
         const data = await response.json();

         if (data.success) {
            showMessage('Merci pour votre avis !', 'success');
            form.reset();
            document.querySelectorAll('.star').forEach(s => s.classList.remove('active'));
         } else {
            showMessage(data.error || 'Erreur', 'error');
         }
      } catch (err) {
         showMessage('Erreur de connexion', 'error');
      }
   });

   function showMessage(text, type) {
      messageResult.textContent = text;
```

```
        messageResult.className = 'message ' + type;
    }
});
```

--- FIN DU CORRIGE SUJET 2 ---