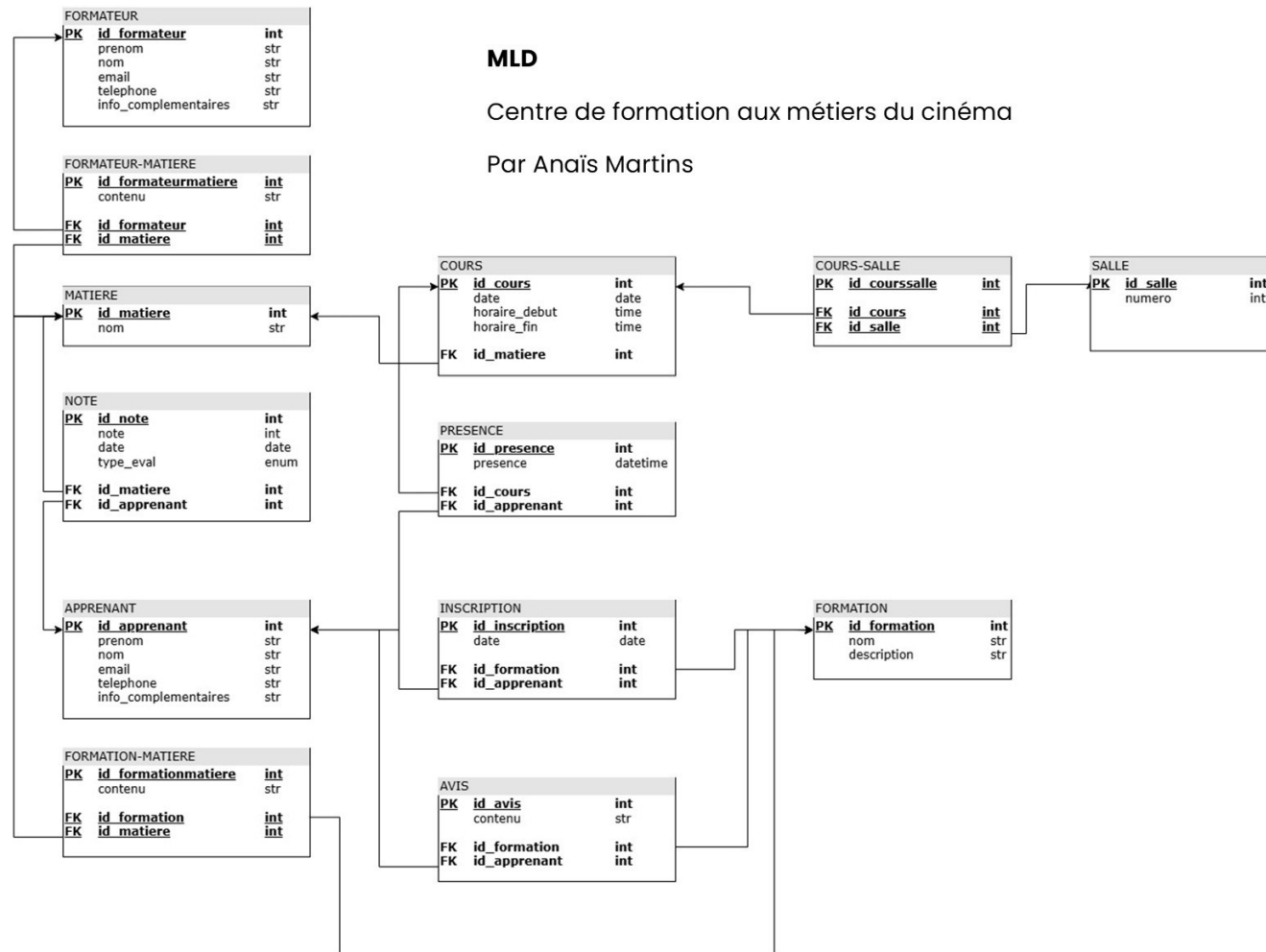
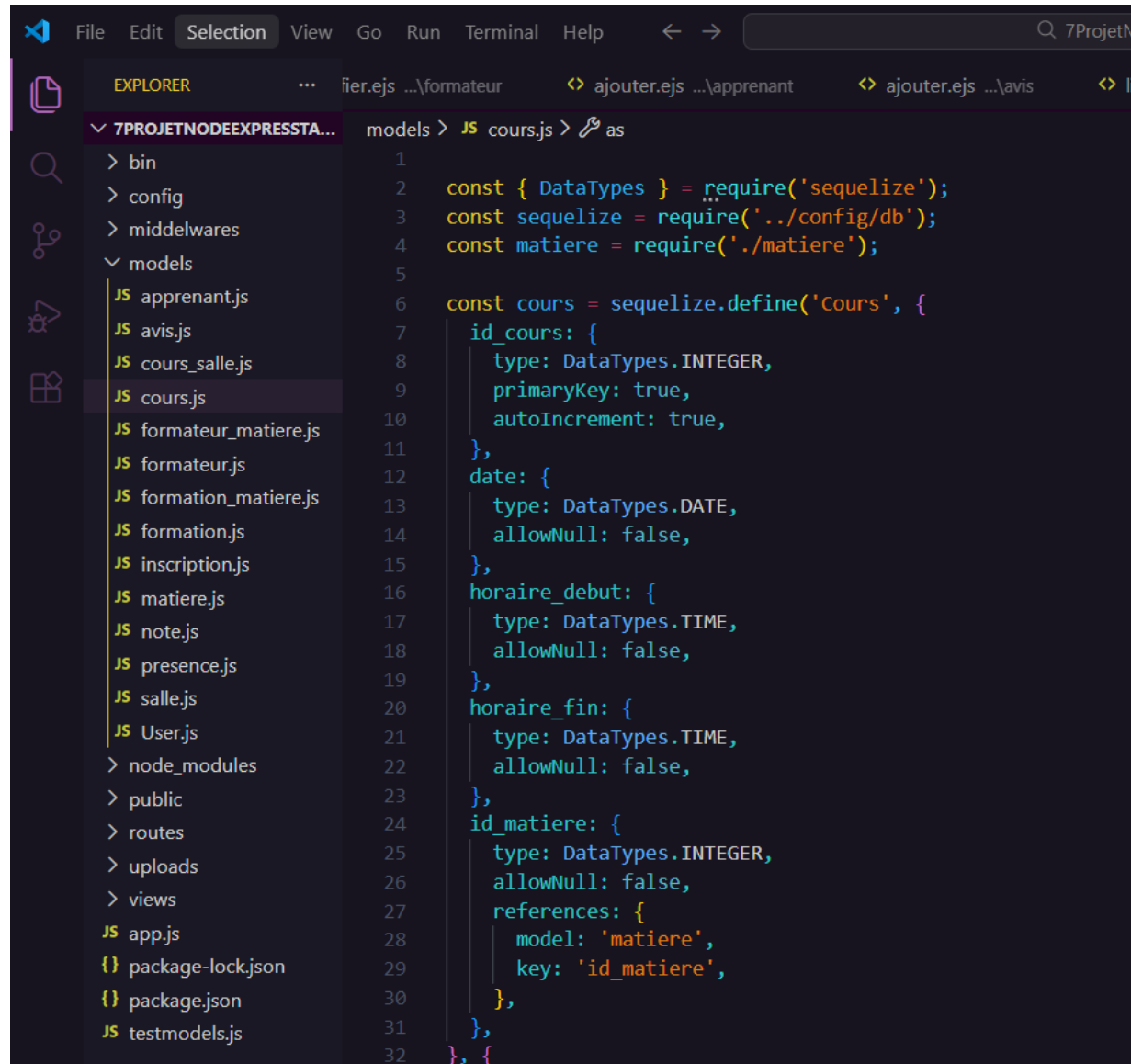


Situation professionnelle # 1

Modèle Logique de Données



Création des modèles

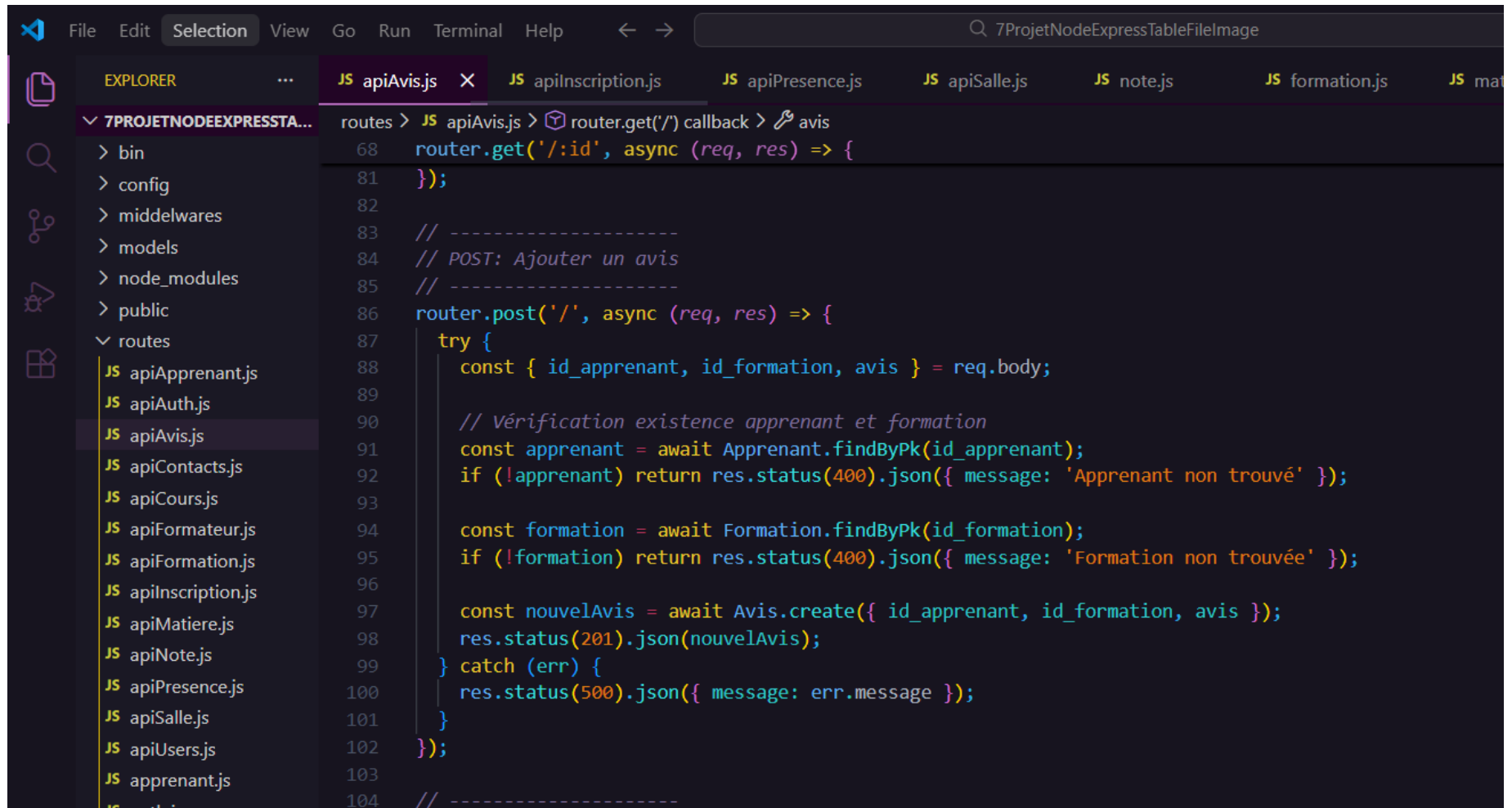


```
File Edit Selection View Go Run Terminal Help 7ProjetN

EXPLORER
7PROJETNODEEXPRESSTA...
  bin
  config
  middlewares
  models
    JS apprenant.js
    JS avis.js
    JS cours_salle.js
    JS cours.js
    JS formateur_matiere.js
    JS formateur.js
    JS formation_matiere.js
    JS formation.js
    JS inscription.js
    JS matiere.js
    JS note.js
    JS presence.js
    JS salle.js
    JS User.js
  node_modules
  public
  routes
  uploads
  views
  JS app.js
  {} package-lock.json
  {} package.json
  JS testmodels.js

models > JS cours.js > as
1
2 const { DataTypes } = require('sequelize');
3 const sequelize = require('../config/db');
4 const matiere = require('../matiere');
5
6 const cours = sequelize.define('Cours', {
7   id_cours: {
8     type: DataTypes.INTEGER,
9     primaryKey: true,
10    autoIncrement: true,
11  },
12  date: {
13    type: DataTypes.DATE,
14    allowNull: false,
15  },
16  horaire_debut: {
17    type: DataTypes.TIME,
18    allowNull: false,
19  },
20  horaire_fin: {
21    type: DataTypes.TIME,
22    allowNull: false,
23  },
24  id_matiere: {
25    type: DataTypes.INTEGER,
26    allowNull: false,
27    references: {
28      model: 'matiere',
29      key: 'id_matiere',
30    },
31  },
32 }, {
```

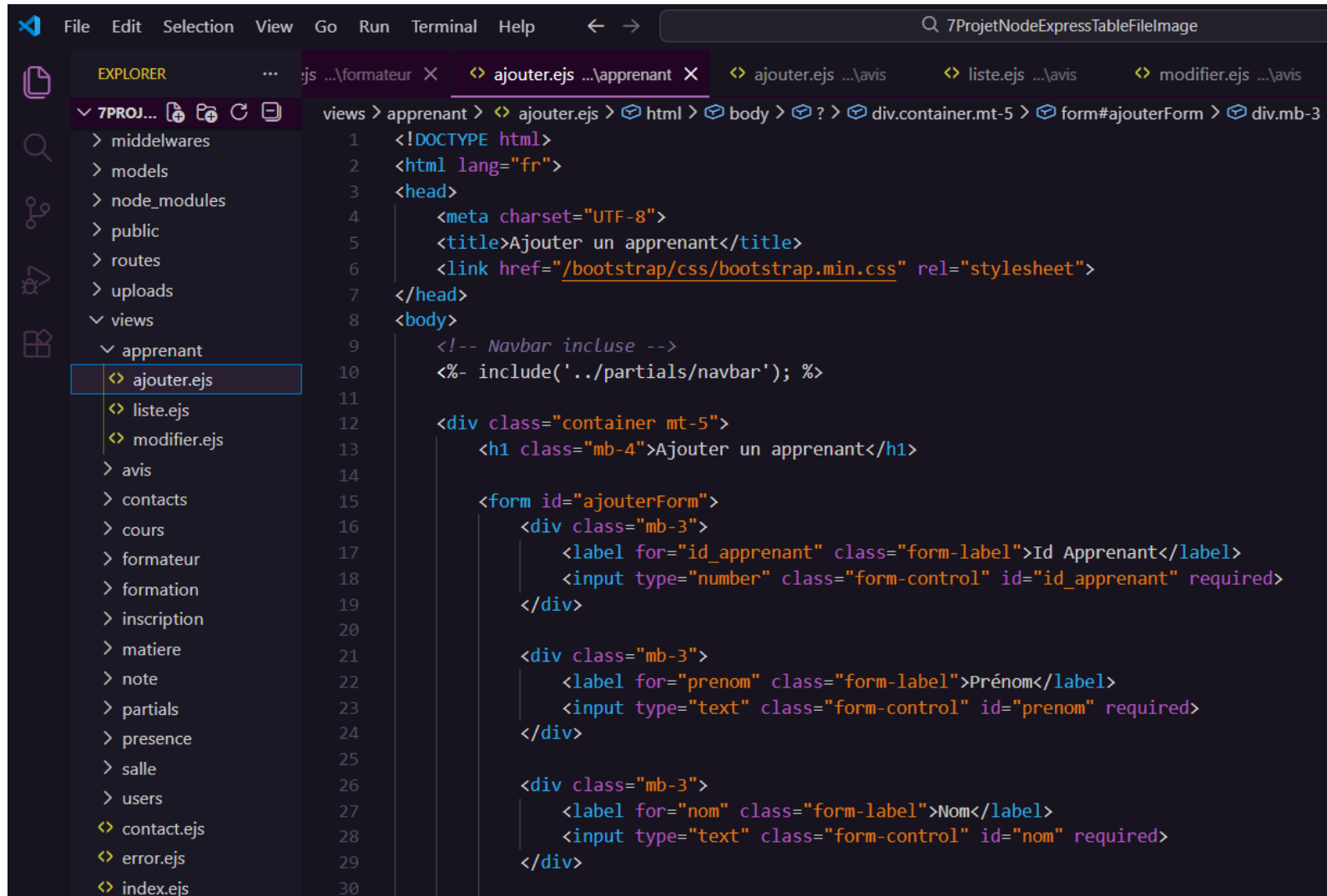
Création des routes



The screenshot shows the Visual Studio Code interface with a project named '7PROJETNODEEXPRESSTA...'. The Explorer panel on the left shows the file structure, with the 'routes' folder expanded. The file 'apiAvis.js' is selected. The main editor displays the code for 'apiAvis.js', which includes a GET route for '/:id' and a POST route for '/'. The code is as follows:

```
68 router.get('/:id', async (req, res) => {
81 });
82
83 // -----
84 // POST: Ajouter un avis
85 // -----
86 router.post('/', async (req, res) => {
87   try {
88     const { id_apprenant, id_formation, avis } = req.body;
89
90     // Vérification existence apprenant et formation
91     const apprenant = await Apprenant.findByPk(id_apprenant);
92     if (!apprenant) return res.status(400).json({ message: 'Apprenant non trouvé' });
93
94     const formation = await Formation.findByPk(id_formation);
95     if (!formation) return res.status(400).json({ message: 'Formation non trouvée' });
96
97     const nouvelAvis = await Avis.create({ id_apprenant, id_formation, avis });
98     res.status(201).json(nouvelAvis);
99   } catch (err) {
100     res.status(500).json({ message: err.message });
101   }
102 });
103
104 // -----
```

Création des views



```
1 <!DOCTYPE html>
2 <html lang="fr">
3 <head>
4   <meta charset="UTF-8">
5   <title>Ajouter un apprenant</title>
6   <link href="/bootstrap/css/bootstrap.min.css" rel="stylesheet">
7 </head>
8 <body>
9   <!-- Navbar include -->
10  <%- include('../partials/navbar'); %>
11
12  <div class="container mt-5">
13    <h1 class="mb-4">Ajouter un apprenant</h1>
14
15    <form id="ajouterForm">
16      <div class="mb-3">
17        <label for="id_apprenant" class="form-label">Id Apprenant</label>
18        <input type="number" class="form-control" id="id_apprenant" required>
19      </div>
20
21      <div class="mb-3">
22        <label for="prenom" class="form-label">Prénom</label>
23        <input type="text" class="form-control" id="prenom" required>
24      </div>
25
26      <div class="mb-3">
27        <label for="nom" class="form-label">Nom</label>
28        <input type="text" class="form-control" id="nom" required>
29      </div>
30    </form>
  
```

Gestion de la Base De Données via PhpMyAdmin

The screenshot displays the phpMyAdmin web interface. On the left, a sidebar shows a tree view of the database structure, including a new database 'centre_formation' and several tables like 'apprenant', 'avis', 'cours', 'cours_salle', 'formateur', 'formateur_matiere', 'formation', 'formation_matiere', 'inscription', 'matiere', 'note', 'presence', and 'salle'. The main area is titled 'Serveur : 127.0.0.1 » Base de données : centre_formation » Table : formateur'. It features a navigation bar with tabs for 'Parcourir', 'Structure', 'SQL', 'Rechercher', 'Insérer', 'Exporter', 'Importer', 'Privileges', 'Opérations', 'Suivi', and 'Déclencheurs'. Below this, there are buttons for 'Structure de table' and 'Vue relationnelle'. The central part of the interface shows a table structure for 'formateur' with the following columns:

#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra	Action
<input type="checkbox"/>	1 id_formateur	int(20)			Non	Aucun(e)		AUTO_INCREMENT	Modifier Supprimer Plus
<input type="checkbox"/>	2 prenom	varchar(50)	utf8mb4_general_ci		Non	Aucun(e)			Modifier Supprimer Plus
<input type="checkbox"/>	3 nom	varchar(100)	utf8mb4_general_ci		Non	Aucun(e)			Modifier Supprimer Plus
<input type="checkbox"/>	4 email	varchar(70)	utf8mb4_general_ci		Non	Aucun(e)			Modifier Supprimer Plus
<input type="checkbox"/>	5 telephone	varchar(20)	utf8mb4_general_ci		Non	Aucun(e)			Modifier Supprimer Plus
<input type="checkbox"/>	6 info_complementaires	varchar(300)	utf8mb4_general_ci		Oui	NULL			Modifier Supprimer Plus

Below the table structure, there are options to 'Ajouter à la liste centrale' or 'Supprimer de la liste centrale de colonnes'. At the bottom, there is a section for 'Ajouter' a new column, with a dropdown menu set to 'après info_complementaires' and an 'Exécuter' button.