REPUBLIC OF CAMEROON

Peace – Work – Fatherland

UNIVERSITY OF DOUALA

REPUBLIQUE DU CAMEROUN

Paix – Travail – Patrie

UNIVERSITE DE DOUALA

INF366 :

**ADMINISTRATION DES BASES DE DONNEES**

THEME:

**APPLICATION DE GESTION D’UN PRESSING**

|  |  |
| --- | --- |
| **NOM ET PRENOM** | **MATRICULE** |
| NGUEFA DJIFACK HERVE STEPHANE | 22S74319 |
| MAKANI LIZA BERENICE CONSTANCE | 21S56946 |
| TCHOUTEZO TCHINDA CARMEL LEROY | 21S57332 |
|  |  |

**Année académique : 2024/2025**

**Superviser par : Mme NGO JOB**

**Groupe : 20**

Table des matières

[INTRODUCTION 3](#_Toc198085511)

[I- ANALYSE DES BESOINS 4](#_Toc198085512)

[II- MODELISATION CONCEPTUELLE 5](#_Toc198085513)

[III- MODELISATION LOGIQUE ET PHYSIQUE 6](#_Toc198085514)

IV- IMPLEMENTATION DE LA BASE DE DONNEES ………………………………… 8

V- TEST ET VALIDATION ………………………………………………………………… 9

VI- CONCLUSION ET PESPECTIVE …………………………………………………… 11

# INTRODUCTION

Avec l’évolution constante des technologies de l’information, de nombreux secteurs d’activité adoptent des solutions informatiques pour améliorer leur fonctionnement et leur efficacité. Le secteur du pressing, bien qu’artisanal à ses débuts, est de plus en plus impacté par cette transformation numérique. La gestion manuelle des commandes, des clients et des articles entraîne souvent des erreurs, des pertes de temps et une insatisfaction des clients. Face à ce constat, la mise en place d’une application dédiée à la gestion des activités d’un pressing s’avère indispensable. Une telle solution permettrait non seulement d’optimiser la gestion quotidienne, mais aussi de garantir un meilleur suivi des opérations, une traçabilité des vêtements confiés, et une meilleure expérience client. Ce projet vise à concevoir et développer une application numérique simple, intuitive et efficace, destinée à automatiser les principales opérations d’un pressing. L'application, développée avec **Streamlit** pour une interface utilisateur interactive et conviviale, repose sur une base de données **SQLite** qui assure une gestion structurée et fiable des données. Elle permet notamment :

* **D’enregistrer et de suivre les commandes,**
* **De créer et gérer les fiches clients,**
* **De gérer les stocks et les articles,**
* **De suivre les paiements et le statut financier des commandes,**
* **D’organiser les informations sur les employés et les opérations quotidiennes.**

Cette solution numérique a pour objectif de réduire les erreurs, d’optimiser la gestion interne du pressing, et de fournir une vue d’ensemble claire grâce à un tableau de bord. Elle s’adresse principalement aux pressings locaux qui souhaitent améliorer leur efficacité sans recourir à des systèmes coûteux ou complexes.

## ANALYSE DES BESOINS

Dans l’optique de produit une solution satisfaisante, nous avons définir :

**Utilisateurs du système :**

* **Administrateur**

**Exigences fonctionnelles :**

* Gestion des clients, commandes, employés, paiements, stock
* Calcul automatique des totaux et points de fidélité

**Exigences non fonctionnelles :**

* Rapidité
* Portabilité (utilisation de Streamlit et SQLite)
* Sécurité basique par authentification

**Règles de gestion :**

* Un client peut avoir plusieurs commandes
* Une commande appartient à un seul client
* Un paiement est lié à une commande
* Plusieurs articles dans une commande

# MODELISATION CONCEPTUELLE

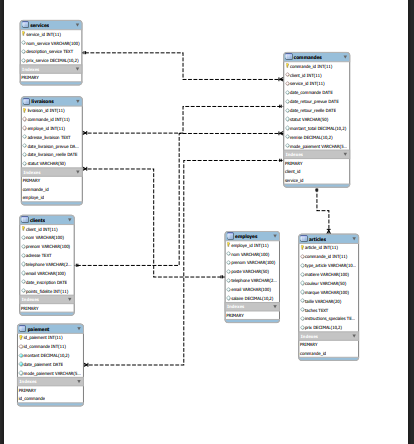
Un diagramme entité-association a été conçu avec les entités suivantes : Client, Commande, Employé, Article, Stock, Paiement. Chaque entité possède des attributs spécifiques avec des contraintes comme les clés primaires et étrangères.

# III- MODELISATION LOGIQUE ET PHYSIQUE

1. **Modèle Logique de Données (MLD)**

Voici la représentation relationnelle du modèle de données :

* **Client** (id\_client, nom, téléphone, adresse, points)
* **Employé** (id\_employé, nom, prénom, poste)
* **Commande** (id\_commande, date\_commande, date\_retour, statut, montant\_total, mode\_paiement, description\_c, #id\_client)
* **Article** (id\_article, description, type\_service, prix\_unitaire, quantité, #id\_commande)
* **Paiement** (id\_paiement, date\_paiement, montant, mode\_paiement, #id\_commande)
* **Service** (Id\_service, nom\_service, description\_service, prix\_service)
* **Livraison** (Id\_livraison, adress, #Id\_employer, #Id\_commande)



Un fichier PDF (modeleLogiquePlusVisible.pdf) est disponible dans les fichier pour une meilleur observation.

* 1. **Script de création de la base de données**

CREATE TABLE "Clients" (

    "client\_id" INTEGER,

    "nom"   TEXT,

    "prenom"    TEXT,

    "adresse"   TEXT,

    "telephone" TEXT,

    "email" TEXT,

    "date\_inscription"  DATE,

    "points\_fidelite"   INTEGER DEFAULT 0,

    PRIMARY KEY("client\_id" AUTOINCREMENT)

);

CREATE TABLE "Services" (

    "service\_id"    INTEGER,

    "nom\_service"   TEXT,

    "description\_service"   TEXT,

    "prix\_service"  REAL,

    PRIMARY KEY("service\_id" AUTOINCREMENT)

);

CREATE TABLE "Employes" (

    "employe\_id"    INTEGER,

    "nom"   TEXT,

    "prenom"    TEXT,

    "poste" TEXT,

    "telephone" TEXT,

    "email" TEXT,

    "salaire"   REAL,

    PRIMARY KEY("employe\_id" AUTOINCREMENT)

);

CREATE TABLE "Livraisons" (

    "livraison\_id"  INTEGER,

    "commande\_id"   INTEGER,

    "employe\_id"    INTEGER,

    "adresse\_livraison" TEXT,

    "date\_livraison\_prevue" DATE,

    "date\_livraison\_reelle" DATE,

    "statut"    TEXT,

    PRIMARY KEY("livraison\_id" AUTOINCREMENT),

    FOREIGN KEY("commande\_id") REFERENCES "Commandes"("commande\_id"),

    FOREIGN KEY("employe\_id") REFERENCES "Employes"("employe\_id")

);

CREATE TABLE "Paiement" (

    "id\_paiement"   INTEGER,

    "id\_commande"   INTEGER,

    "montant"   REAL NOT NULL,

    "date\_paiement" TEXT NOT NULL,

    "mode\_paiement" TEXT,

    PRIMARY KEY("id\_paiement" AUTOINCREMENT),

    FOREIGN KEY("id\_commande") REFERENCES "Commande"("id\_commande")

);

CREATE TABLE "Articles" (

    "article\_id"    INTEGER,

    "commande\_id"   INTEGER,

    "type\_article"  TEXT,

    "matiere"   TEXT,

    "couleur"   TEXT,

    "marque"    TEXT,

    "taille"    TEXT,

    "taches"    TEXT,

    "instructions\_speciales"    TEXT,

    "prix"  REAL,

    PRIMARY KEY("article\_id" AUTOINCREMENT),

    FOREIGN KEY("commande\_id") REFERENCES "Commandes"("commande\_id")

);

CREATE TABLE "Commandes" (

    "commande\_id"   INTEGER,

    "client\_id" INTEGER,

    "service\_id"    INTEGER,

    "date\_commande" DATE,

    "date\_retour\_prevue"    DATE,

    "date\_retour\_reelle"    DATE,

    "statut"    TEXT,

    "montant\_total" REAL,

    "remise"    REAL,

    "mode\_paiement" TEXT,

    PRIMARY KEY("commande\_id" AUTOINCREMENT),

    FOREIGN KEY("client\_id") REFERENCES "Clients"("client\_id")

    FOREIGN KEY("service\_id") REFERENCES "Services"("service\_id")

);

**IV- IMPLEMENTATION DE LA BASE DE DONNEES**

* + 1. **LANGAGE DE MANIPULATION DES DONNEES**
* **INSERTION DES DONNEES**
* Insertion d’un nouveau service :

**INSERT INTO Services (nom\_service, prix\_service, description\_service) VALUES (?, ?, ?)**

* Insertion d'un nouveau client:

**INSERT INTO Clients (nom, prenom, adresse, telephone, email, date\_inscription)**

**VALUES (?, ?, ?, ?, ?, ?)**

* Insertion de la commande :

**INSERT INTO Commandes (client\_id, date\_commande, date\_retour\_prevue, montant\_total, remise, adress\_livraison, statut) VALUES (?, ?, ?, ?, ?, ?, ?)**

* Insertion des articles de la commande :

**INSERT INTO Articles (commande\_id, type\_article, matiere, couleur, marque, taille)**

**VALUES (?, ?, ?, ?, ?, ?)**

* **SELECTION DE DONNEES**
* Total de commandes :

**SELECT COUNT(\*) FROM Commandes**

* Commandes en cours :

**SELECT COUNT(\*) FROM Commandes WHERE statut = 'En cours'**

* Chiffre d'affaires total :

**SELECT (SUM(montant\_total)) FROM Commandes**

* **MODIFICATION DE DONNEES**
* Mise à jour des points de fidélité d'un client existant**:**

**UPDATE Clients SET points\_fidelite = points\_fidelite + 1 WHERE client\_id = ?**

* + 1. **REQUETTE DE CONSULTATION AVEC CONDITION**
* **SELECT c.commande\_id, cl.adresse**

**FROM Commandes c**

**JOIN Clients cl ON c.client\_id = cl.client\_id**

**LEFT JOIN Livraisons l ON c.commande\_id = l.commande\_id**

**WHERE c.statut = 'Terminé' AND l.commande\_id IS NULL;**

**NB :** Cette requête permet de récupérer les commandes dont le statut est "Terminé" mais qui n'ont pas encore de livraison enregistrée, elle utilise une jointure avec la table Livraisons pour vérifier si une livraison a déjà été effectuée pour la commande. Si ce n'est pas le cas, la commande est affichée pour être livrée.

* **SELECT employe\_id AS ID, nom AS Nom**

**FROM Employes**

**WHERE poste = 'Livreur'**

**ORDER BY nom**

**NB :** Cette requête permet de récupérer la liste des livreurs disponibles dans la table Employes

* **SELECT c.commande\_id, cl.nom, cl.prenom, c.date\_commande, c.date\_retour\_prevue, c.statut, c.remise, c.montant\_total**

**FROM Commandes c**

**JOIN Clients cl ON c.client\_id = cl.client\_id**

**ORDER BY c.date\_commande DESC**

**NB:** Cette requête permet de récupérer les informations de toutes les commandes (commande\_id, date\_commande, date\_retour\_prevue, etc.) ainsi que les informations du client associé (nom et prenom), elle joint les tables Commandes et Clients en utilisant le client\_id.

* **SELECT c.commande\_id, cl.nom, cl.prenom, c.date\_commande,**

**c.date\_retour\_prevue, c.statut, c.remise, c.montant\_total**

**FROM Commandes c**

**JOIN Clients cl ON c.client\_id = cl.client\_id**

**ORDER BY c.date\_commande DESC**

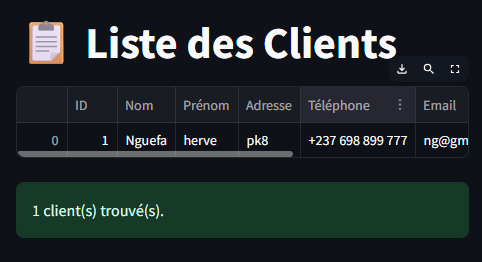
**NB** : Cela permettrait de récupérer les commandes récentes en priorité.

**V- TEST ET VALIDATION**

* + - **Ajoute d’un service**
* "INSERT INTO Services (nom\_service, prix\_service, description\_service) VALUES (?, ?, ?)",(nom\_service, prix\_service, description)

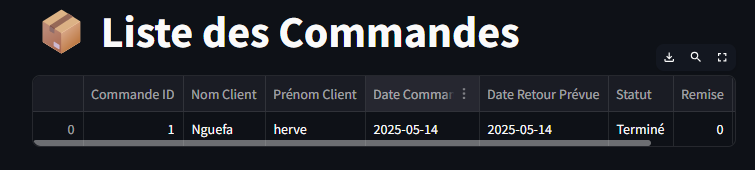
****

* 1. **Ajout d’un client**
* INSERT INTO Clients (nom, prenom, adresse, telephone, email, date\_inscription) VALUES (?, ?, ?, ?, ?, ?), (nom, prenom, adresse, telephone, email, datetime.now().strftime('%Y-%m-%d')))

****

* 1. **Ajout d’une commande**

1. INSERT INTO Commandes (client\_id, date\_commande, date\_retour\_prevue, montant\_total, remise, adress\_livraison, statut) VALUES (?, ?, ?, ?, ?, ?, ?), (client\_id, date\_commande, date\_retour\_prevue, montant\_total, remise, adresse\_livraison, statut\_commande))

****

* **Ajouter un employer**
* INSERT INTO Employes (nom, prenom, poste, telephone,email,salaire) VALUES (?, ?, ?, ?, ?, ?),(nom, prenom, poste, telephone, email, salaire)



**VI- Conclusion et Perspectives**

Le développement de cette application Streamlit de gestion de pressing a permis de mettre en place une solution complète et intuitive pour la gestion des opérations d’un pressing. Le système couvre les fonctionnalités essentielles telles que :

* L’enregistrement des clients et des commandes,
* La gestion des services et articles associés,
* La facturation automatique avec application de remises,
* Le suivi du traitement,
* La gestion des livraisons avec affectation de livreurs,
* L'affichage de tableaux de bord interactifs pour le suivi des KPI.

Grâce à l’interface conviviale de Streamlit et à l'utilisation d'une base de données relationnelle SQLite, l'application offre un bon compromis entre simplicité et efficacité pour les petites structures cherchant à digitaliser leurs services.

**Limites du projet**

Malgré ces avancées, le projet présente certaines limites :

* Analyse des données limitée : L’application ne dispose pas encore d’un module avancé d’analyse des performances (clients fidèles, services les plus demandés, pics d’activité...).
* Sécurité rudimentaire : L’authentification actuelle est simple et ne prend pas encore en charge les rôles (admin, employé, livreur...).
* Performance des requêtes : Pour de très grands volumes de données, certaines requêtes peuvent devenir lentes, surtout avec SQLite.
* Expérience utilisateur statique : L’interface pourrait être enrichie avec davantage d’animations ou de retours visuels pour une expérience plus dynamique.

**Perspectives d’évolution**

Pour rendre le système encore plus robuste et évolutif, plusieurs pistes d’amélioration sont envisagées :

1. Ajout d’un module d’analyse avancée :
   * Utilisation de bibliothèques comme Plotly ou Seaborn pour des graphiques dynamiques,
   * Création d’indicateurs de performance personnalisés (CA mensuel, taux de fidélité…).
2. Implémentation d’un système de rôles et permissions :
   * Séparer les accès pour les administrateurs, livreurs, et employés simples,
   * Ajouter une traçabilité des actions.
3. Optimisation de la base de données :
   * Passage à une base PostgreSQL pour une meilleure scalabilité,

Notifications et alertes :

* + Intégration d’un système de notification (mail, SMS ou pop-up) pour informer les clients lorsque leur commande est prête ou livrée.

1. Application mobile :
   * Développement d'une version mobile via Streamlit Mobile ou Flutter pour un accès rapide par les clients et les livreurs.
2. Intégration d’un module de paiement :