# RAPPORT DU PROJET TUTORÉ

## ORAFLEX – Chatbot de Support Client Bancaire

| | |
|---|---|
| **Nom** | **GOUBA** |
| **Prénom** | **Georgette** |
| **INE** | **N01455020222** |

1. **Contexte Général du Projet**
   Dans le cadre du module **Projet Tutoré d'Analyse de Données**, les étudiants de 3e année en Génie Logiciel doivent développer un système intégrant collecte, compréhension et exploitation de données à travers des technologies modernes (Python, NLP, Machine Learning, bases de données, etc.).
   Le but de ce projet est de permettre aux étudiants d'appliquer leurs compétences en :

- Programmation Python,
- Développement backend,
- Analyse des données textuelles,
- Modélisation NLP,
- Conception de base de données,
- Création d'interfaces utilisateurs,
- Et documentation technique.

Bien que le thème global du module porte sur l'analyse de données, les projets peuvent inclure des aspects NLP, IA appliquée et traitement automatique du langage — ce qui est le cas du présent projet.
Ce rapport présente le développement du **chatbot bancaire ORAFLEX**, un assistant conversationnel intelligent destiné aux services bancaires de base.

2. **Présentation du Projet**

**Titre du projet:**

Développement d'un Chatbot de Support Client pour les Services Bancaires

**Description :**

Le projet consiste à concevoir un chatbot capable d'interagir automatiquement avec les clients d'une banque. J'ai choisi (ORABANK dans la simulation) pour répondre à leurs questions fréquentes concernant :

- Les comptes,
- Les transactions,
- Les cartes bancaires,
- Les agences,
- Les codes OTP,
- Les virements,
- Le solde,
- L'historique,
- Et de l'aide.

Pour faire un chatbot conforme aux réalités du Burkina Faso j'ai pris le cas de ma banque Orabank. Mon chatbot utilise un modèle NLP entraîné avec SpaCy pour classifier les intentions (intents) des utilisateurs.

3. Objectif Général
   L'objectif général de ce projet est de créer un chatbot intelligent que j'ai nommé **ORAFLEX Chatbot** qui répond automatiquement aux utilisateurs en comprenant leurs questions grâce au NLP, et en leur fournissant des informations bancaires fiables et structurées.

4. **Objectifs Spécifiques**
   - Comprendre les requêtes en langage naturel
   - Classifier les intentions utilisateurs (solde, virement, aide, historique, etc.)
   - Générer des réponses automatiques adaptées
   - Fournir un widget de chatbot intégré dans une interface web
   - Stocker les utilisateurs, messages et interactions dans une base PostgreSQL
   - Éventuellement détecter des anomalies dans les interactions

5. Technologies Utilisées

**Backend**

- **Flask** (Framework web Python)
- **SpaCy** (NLP)
- **Python 3.13.5**

**Frontend**

- **HTML/CSS**
- Widget de chatbot dynamique

**Base de Données**

- **PostgreSQL**
- SQLAlchemy (ORM)

**Outils**

- VS Code
- Google Colab (pour entraîner le modèle)
- SpaCy CLI

## 6. Architecture Générale du Système

Le système ORAFLEX est composé de 4 parties :

Interface Web (Chatbot UI)

Serveur Flask (Gestion des requêtes et réponses)

Modèle NLP (SpaCy – Classification des Intentions)

Base de données PostgreSQL (Clients, interactions, logs)

## 7. Modèle NLP

### 7.1. Objectif

Entraîner un modèle SpaCy TextCat pour reconnaître automatiquement les catégories (intentions) des messages utilisateurs.

### 7.2. Intentions utilisées

Le modèle final comprend les labels suivants :

- Salutation
- Solde_compte
- Historique_transactions
- Faire_virement
- Depot_argent

- Retrait_argent
- Code_otp
- Blocage_carte
- Localisation_agence
- Frais_bancaires
- Probleme_application
- Service_client
- Ouverture_compte

  Ajoutés :
- Solde
- Virement
- Historique
- Aide

  **7.3. Entraînement**
- Dataset manuel seulement TextCat
- Utilisation de **train_data.py** + **prepare_data.py** sauvegardé dans "C:\Users\Pc\Desktop\DATA ANALYSE\CHATBOT_BANQUE\model
- Sauvegarde du modèle dans "C:\Users\Pc\Desktop\DATA ANALYSE\CHATBOT_BANQUE\model\model_orabank"
- meta.json modifié

## 8. Base de Données

### 8.1. Modèle Client

```python
# --- Table Client ---
class Client(db.Model):
    __tablename__ = 'client'
    id_client = db.Column(db.Integer, primary_key=True)
    prenom = db.Column(db.String(50), nullable=False)
    nom = db.Column(db.String(50), nullable=False)
    email = db.Column(db.String(120), unique=True, nullable=False)
    password = db.Column(db.String(200), nullable=False)
```

### 8.2. Purpose

Servir à :
- Créer des comptes utilisateurs,
- Stocker credentials,
- Logger les conversations.

9. Backend Flask

   Mon **app.py final** gère :

- Chargement du modèle SpaCy
- Classification des intentions
- Réponse dynamique (random choice)
- Enregistrement en base
- Affichage sur interface web

10. Frontend

Mon chatbot inclut :

- ✓ **Une interface web simple et responsive**, développée en HTML, CSS et Flask.

- ✓ **Un widget de discussion fonctionnel**, permettant d'afficher :

Les messages envoyés par l'utilisateur,

Les réponses générées automatiquement par le chatbot.

- ✓ **Une barre d'actions rapide** située en bas de la fenêtre du chatbot, contenant :

**Solde**
**Virement**
**Historique**
**Aide**

- ✓ **Un modèle NLP personnalisé**, entraîné pour reconnaître plusieurs intentions (ex : solde, dépôt, virement, historique, carte, fraude, etc.).

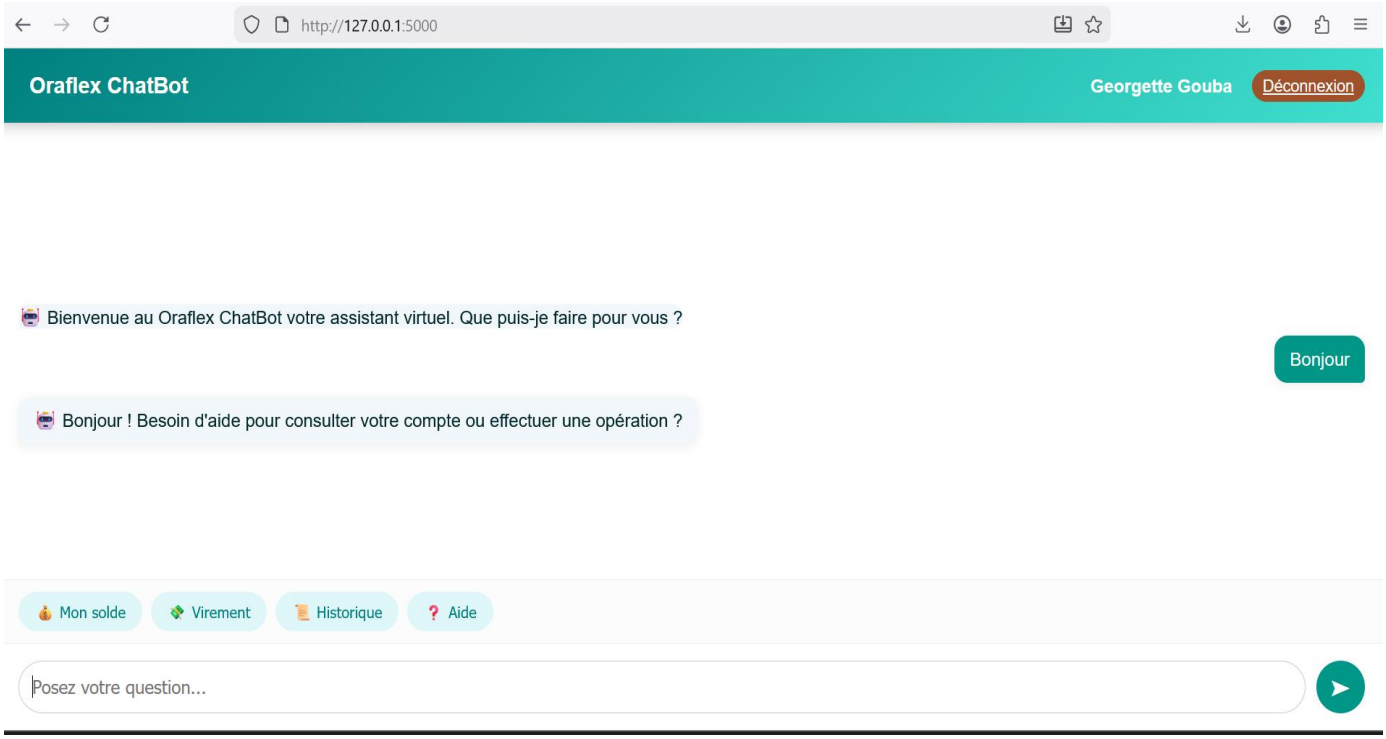
- ✓ **Une logique de traitement backend** permettant d'interpréter les messages et renvoyer une réponse adaptée.
- ✓ **Une base de données PostgreSQL** pour stocker les :

Utilisateurs,
Interactions,
Logs conversationnels (horodatage + messages).

**11. Des Captures d'Écran illustratives**

**A. <u>Interface & Design</u>**

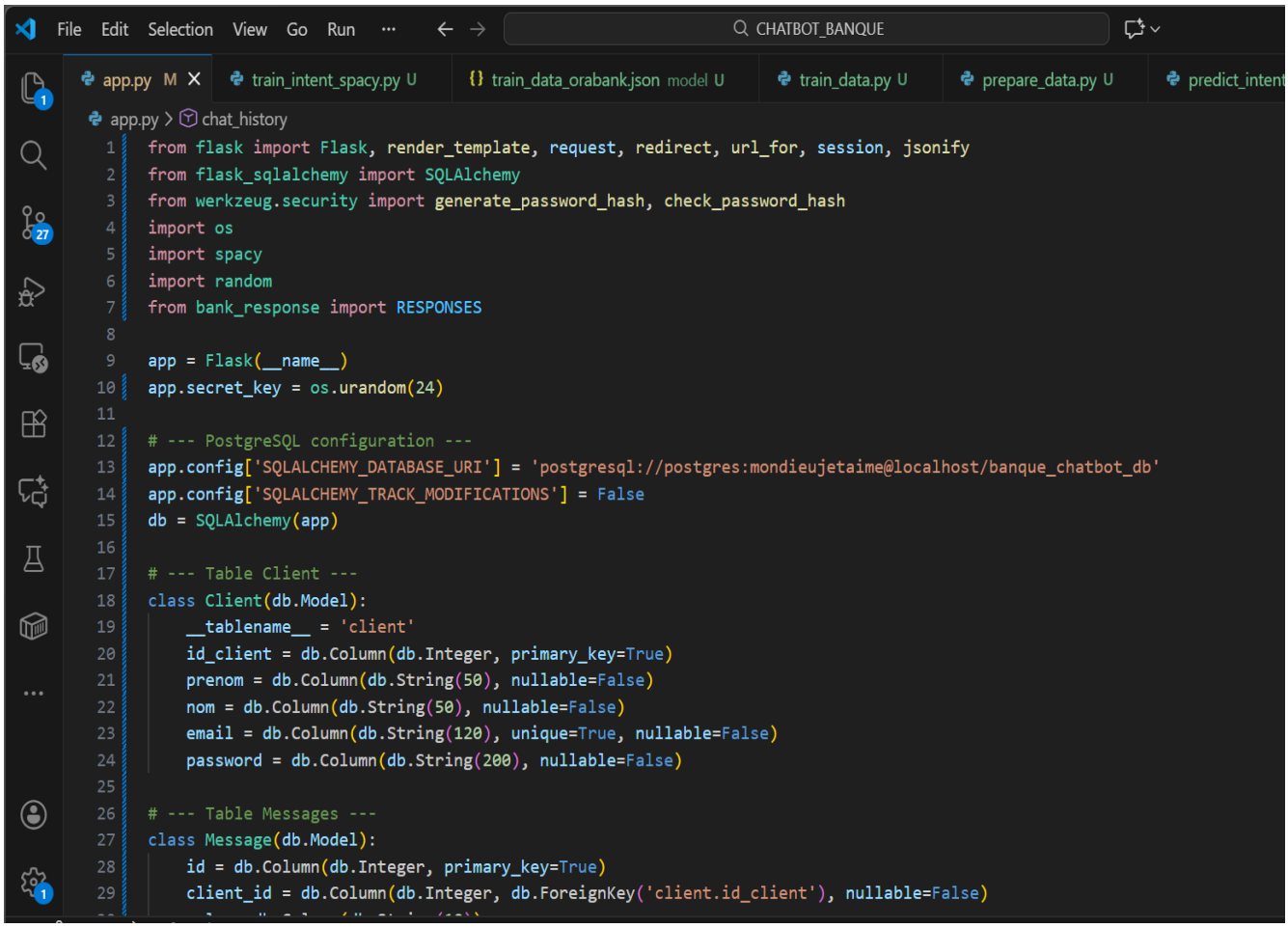
**2. Page du chatbot en fonctionnement (vue complète)**



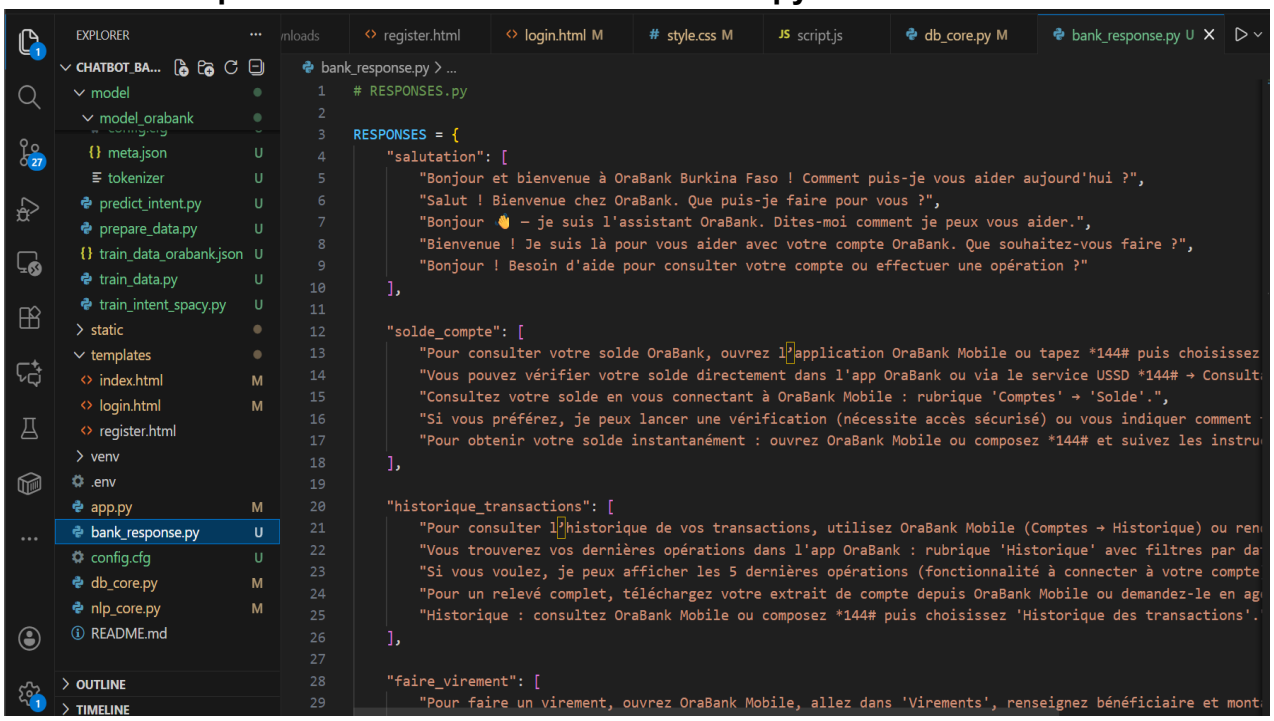

3. **La zone des catégories en bas (solde, virement, historique, aide)**


- **B. <u>Backend</u>**

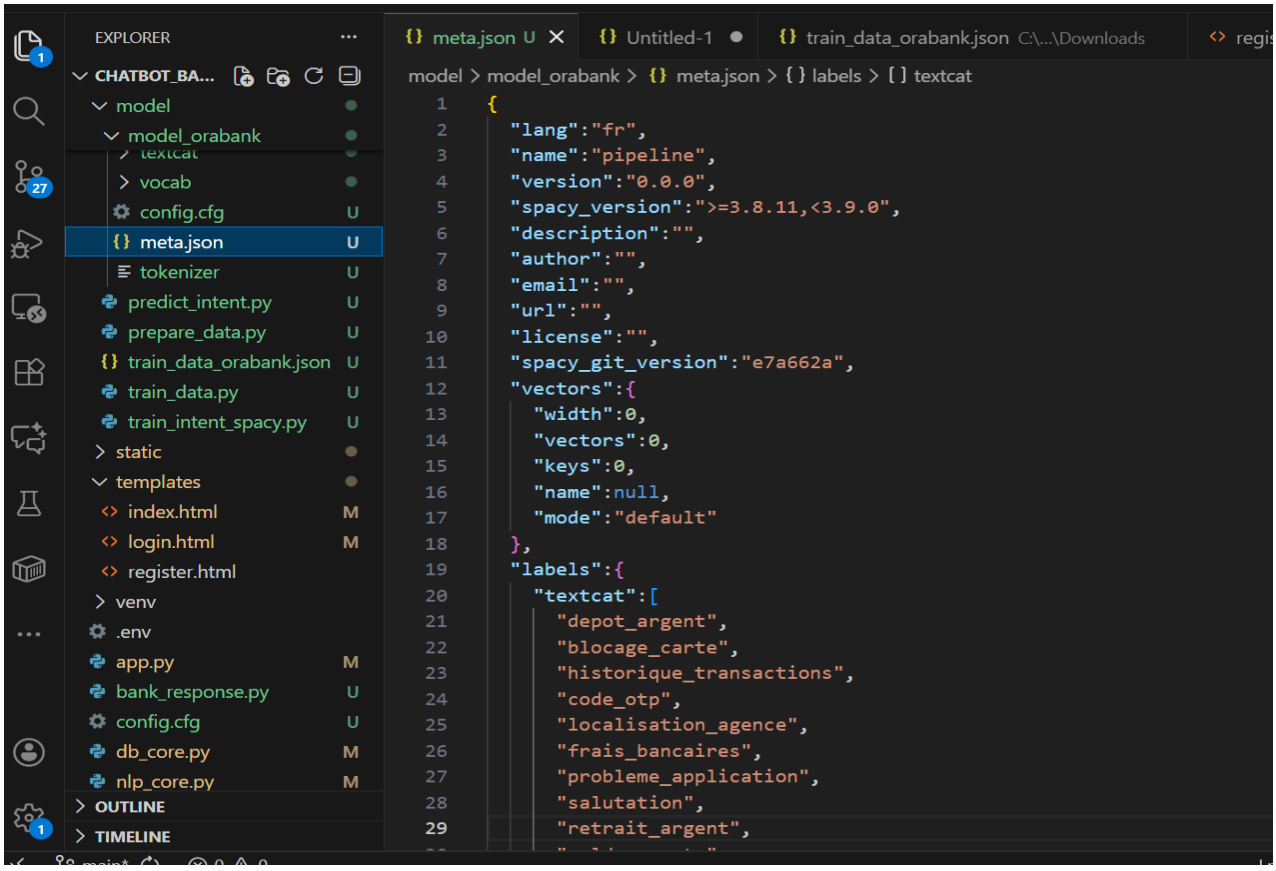
- **Une portion de mon fichier app.py dans VS Code**




- **Une portion de mon fichier RESPONSES.py**

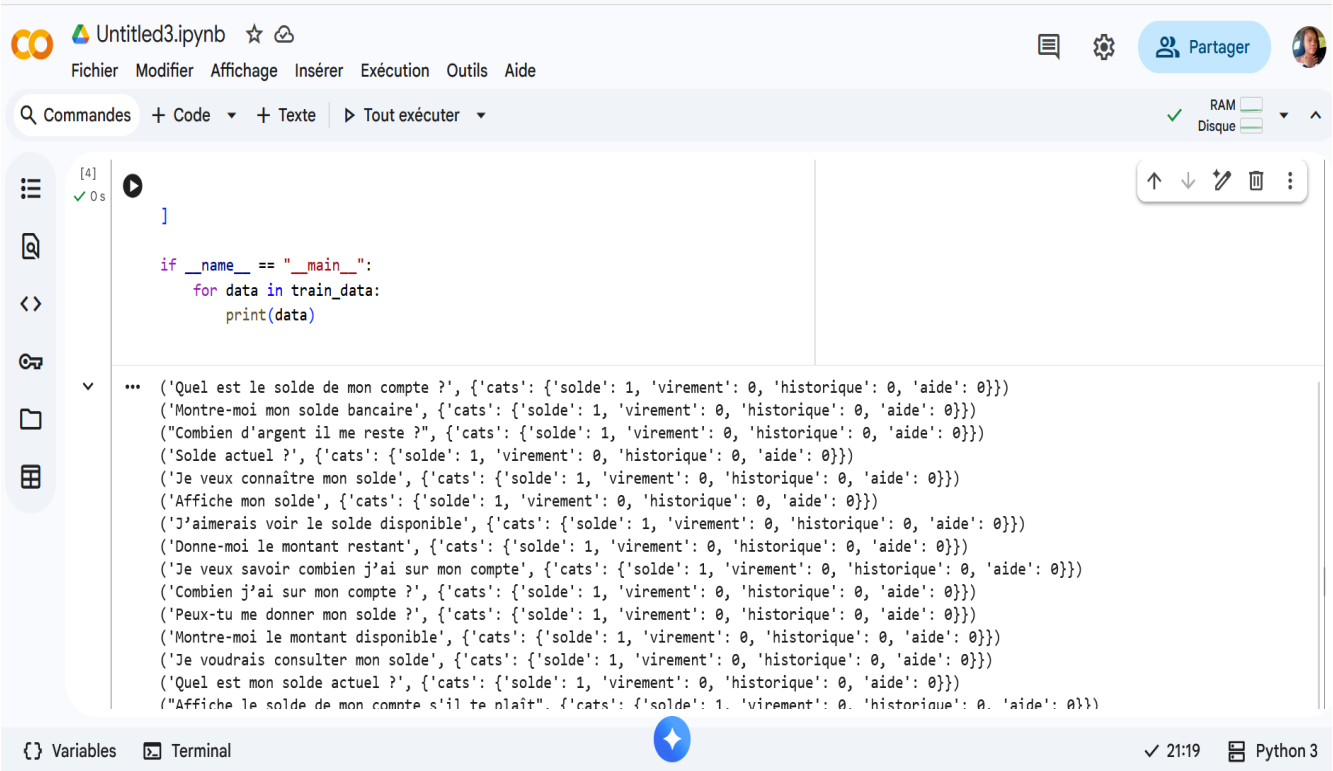


- **Une portion de mon fichier model.json/meta.json du modèle SpaCy**



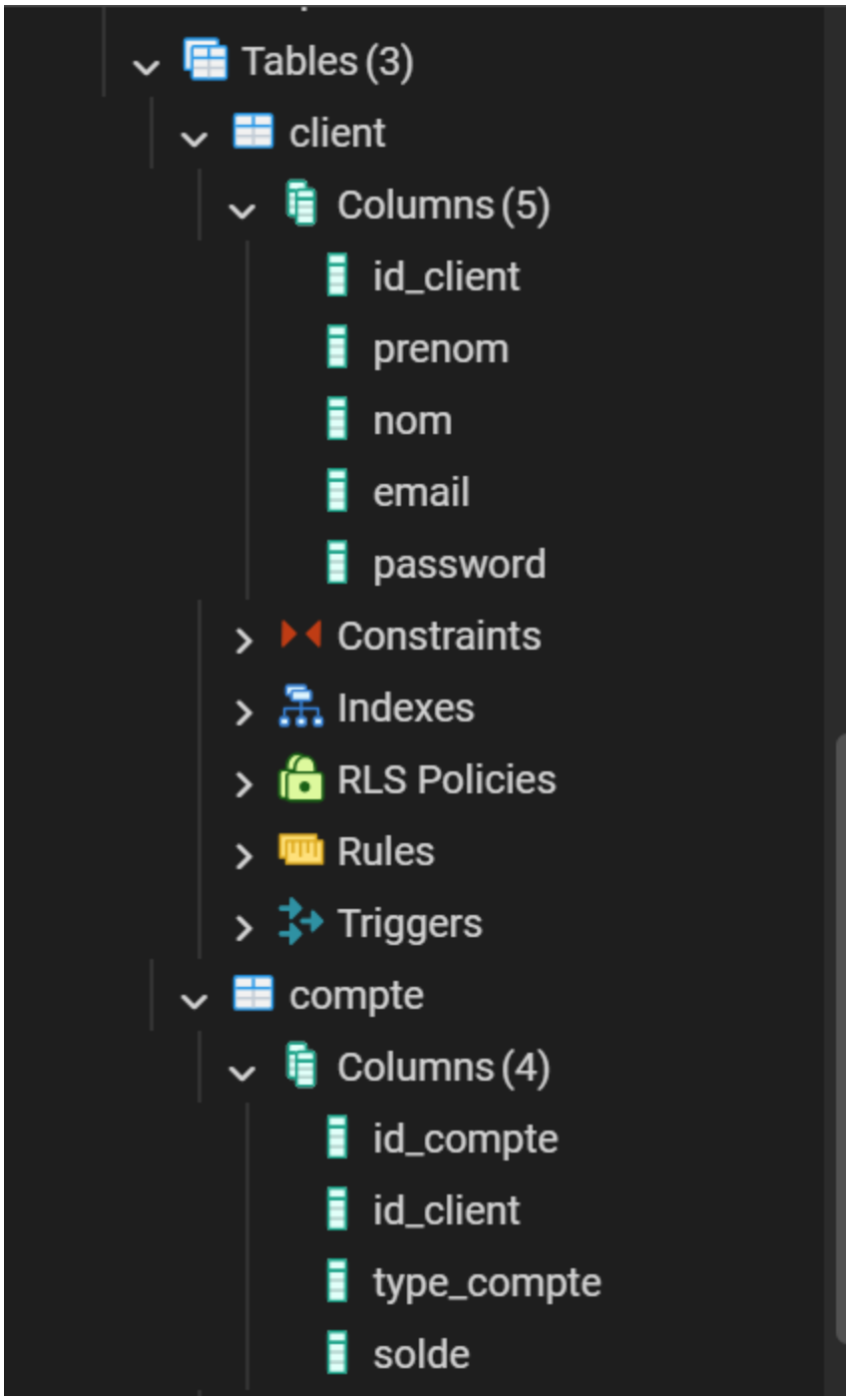

- **<u>C. NLP</u>**

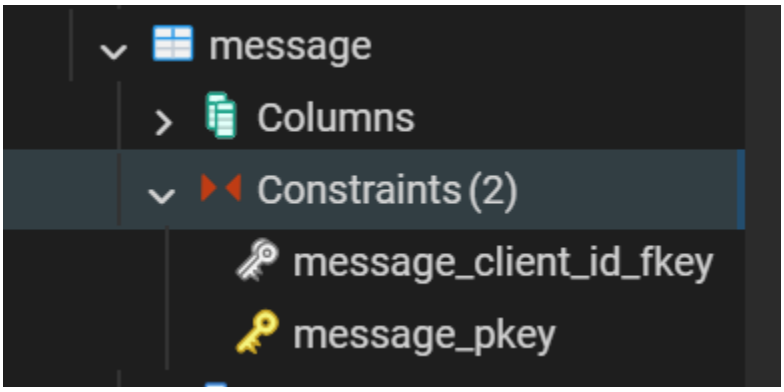
- **Mon Google colab montrant l'entraînement du modèle SpaCy**




## D. Base de Données

- **Les Tables (client, compte et message)**


Tables (3)
client
Columns (5)
id_client
prenom
nom
email
password
Constraints
Indexes
RLS Policies
Rules
Triggers
compte
Columns (4)
id_compte
id_client
type_compte
solde




- **E. Fonctionnement**

- **Exemples de conversations réussies**