

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

```
# --- 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. Interface & Design

2. Page du chatbot en fonctionnement (vue complète)

Oraflex ChatBot

Georgette Gouba Déconnexion

🤖 Bienvenue au Oraflex ChatBot votre assistant virtuel. Que puis-je faire pour vous ?

Bonjour

🤖 Bonjour ! Besoin d'aide pour consulter votre compte ou effectuer une opération ?

Mon solde Virement Historique Aide

Posez votre question...



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

Mon solde Virement Historique Aide

Posez votre question...



- **B. Backend**

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

```

File Edit Selection View Go Run ... ← → Q CHATBOT_BANQUE
app.py M X train_intent_spacy.py U {} train_data_orabank.json model U train_data.py U prepare_data.py U predict_intent.py

app.py > chat_history
from flask import Flask, render_template, request, redirect, url_for, session, jsonify
from flask_sqlalchemy import SQLAlchemy
from werkzeug.security import generate_password_hash, check_password_hash
import os
import spacy
import random
from bank_response import RESPONSES

app = Flask(__name__)
app.secret_key = os.urandom(24)

# --- PostgreSQL configuration ---
app.config['SQLALCHEMY_DATABASE_URI'] = 'postgresql://postgres:mondieujetaime@localhost/banque_chatbot_db'
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False
db = SQLAlchemy(app)

# --- 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)

# --- Table Messages ---
class Message(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    client_id = db.Column(db.Integer, db.ForeignKey('client.id_client'), nullable=False)

```

- Une portion de mon fichier RESPONSES.py

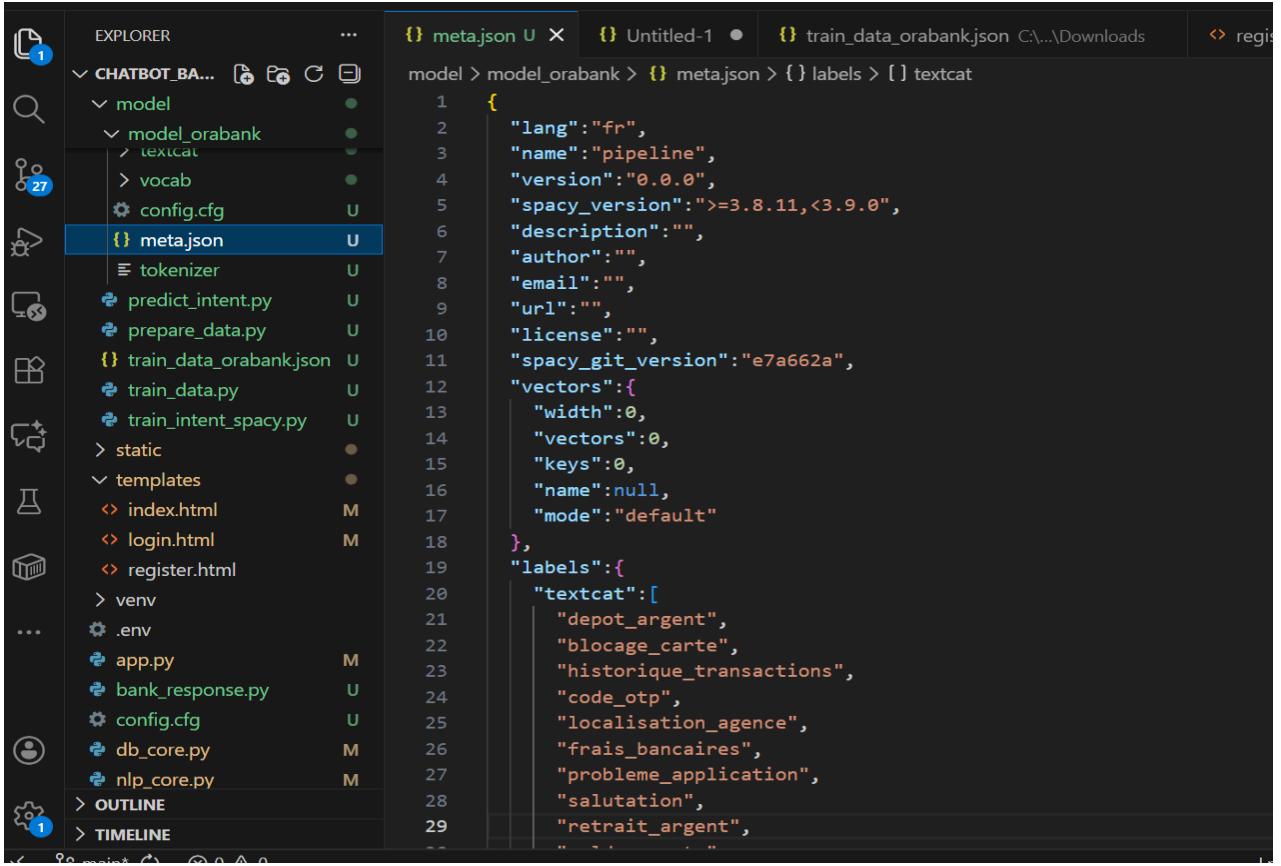
```

EXPLORER ... mloads register.html login.html # style.css M JS script.js db_core.py M bank_response.py U ...
CHATBOT.BA... model model_orabank ...
metajson tokenizer predict.intent.py prepare_data.py train_data_orabank.json train_data.py train_intent_spacy.py static templates index.html login.html register.html venv .env app.py bank_response.py config.cfg db_core.py nlp_core.py README.md

# RESPONSES.py
RESPONSES = {
    "salutation": [
        "Bonjour et bienvenue à OraBank Burkina Faso ! Comment puis-je vous aider aujourd'hui ?",
        "Salut ! Bienvenue chez OraBank. Que puis-je faire pour vous ?",
        "Bonjour - je suis l'assistant OraBank. Dites-moi comment je peux vous aider.",
        "Bienvenue ! Je suis là pour vous aider avec votre compte OraBank. Que souhaitez-vous faire ?",
        "Bonjour ! Besoin d'aide pour consulter votre compte ou effectuer une opération ?"
    ],
    "solde_compte": [
        "Pour consulter votre solde OraBank, ouvrez l'application OraBank Mobile ou tapez *144# puis choisissez",
        "Vous pouvez vérifier votre solde directement dans l'app OraBank ou via le service USSD *144# > Consulte",
        "Consultez votre solde en vous connectant à OraBank Mobile : rubrique 'Comptes' > 'Solde' .",
        "Si vous préférez, je peux lancer une vérification (nécessite accès sécurisé) ou vous indiquer comment",
        "Pour obtenir votre solde instantanément : ouvrez OraBank Mobile ou composez *144# et suivez les instruc"
    ],
    "historique_transactions": [
        "Pour consulter l'historique de vos transactions, utilisez OraBank Mobile (Comptes > Historique) ou ren",
        "Vous trouverez vos dernières opérations dans l'app OraBank : rubrique 'Historique' avec filtres par da",
        "Si vous voulez, je peux afficher les 5 dernières opérations (fonctionnalité à connecter à votre compte",
        "Pour un relevé complet, téléchargez votre extrait de compte depuis OraBank Mobile ou demandez-le en ag",
        "Historique : consultez OraBank Mobile ou composez *144# puis choisissez 'Historique des transactions'."
    ],
    "faire_virement": [
        "Pour faire un virement, ouvrez OraBank Mobile, allez dans 'Virements', renseignez bénéficiaire et mont"
    ]
}

```

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



The screenshot shows a code editor interface with two main panes. The left pane is the 'EXPLORER' view, displaying a file tree for a project named 'CHATBOT_BA...'. The 'meta.json' file is selected and highlighted in blue. The right pane is the 'editor' view, showing the contents of the 'meta.json' file. The file is a JSON object with several properties, including 'lang', 'name', 'version', 'spacy_version', 'description', 'author', 'email', 'url', 'license', 'spacy_git_version', 'vectors', and 'labels'. The 'labels' property contains an array of categories such as 'depot_argent', 'blocage_carte', etc.

```

1   {
2     "lang": "fr",
3     "name": "pipeline",
4     "version": "0.0.0",
5     "spacy_version": ">=3.8.11,<3.9.0",
6     "description": "",
7     "author": "",
8     "email": "",
9     "url": "",
10    "license": "",
11    "spacy_git_version": "e7a662a",
12    "vectors": {
13      "width": 0,
14      "vectors": 0,
15      "keys": 0,
16      "name": null,
17      "mode": "default"
18    },
19    "labels": {
20      "textcat": [
21        "depot_argent",
22        "blocage_carte",
23        "historique_transactions",
24        "code_otp",
25        "localisation_agence",
26        "frais_bancaires",
27        "probleme_application",
28        "salutation",
29        "retrait_argent",
30        "autre"
31      ]
32    }
33  }

```

- C. NLP
- Mon Google colab montrant l'entraînement du modèle SpaCy

The screenshot shows a Jupyter Notebook interface with the following details:

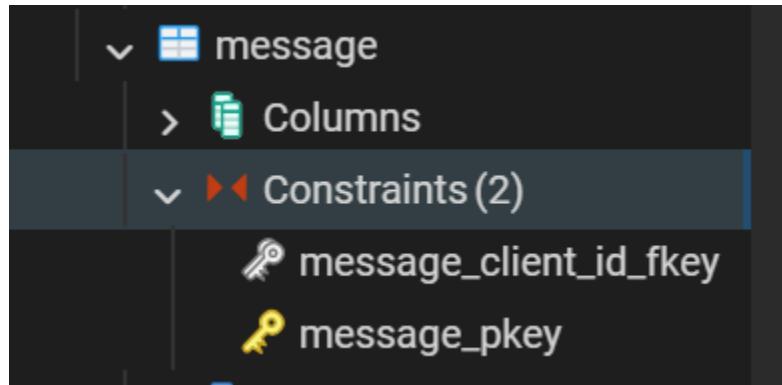
- Title:** Untitled3.ipynb
- Toolbar:** Fichier, Modifier, Affichage, Insérer, Exécution, Outils, Aide.
- Code Cell:** Contains the following Python code:

```
if __name__ == "__main__":
    for data in train_data:
        print(data)
```
- Output:** A list of generated text responses, starting with:
 - 'Quel est le solde de mon compte ?', {'cats': {'solde': 1, 'virement': 0, 'historique': 0, 'aide': 0}})
 - ('Montre-moi mon solde bancaire', {'cats': {'solde': 1, 'virement': 0, 'historique': 0, 'aide': 0}})
 - ("Combien d'argent il me reste ?", {"cats": {"solde": 1, "virement": 0, "historique": 0, "aide": 0}})
 - ('Solde actuel ?', {"cats": {"solde": 1, "virement": 0, "historique": 0, "aide": 0}})
 - ('Je veux connaître mon solde', {"cats": {"solde": 1, "virement": 0, "historique": 0, "aide": 0}})
 - ('Affiche mon solde', {"cats": {"solde": 1, "virement": 0, "historique": 0, "aide": 0}})
 - ('J'aimerais voir le solde disponible', {"cats": {"solde": 1, "virement": 0, "historique": 0, "aide": 0}})
 - ('Donne-moi le montant restant', {"cats": {"solde": 1, "virement": 0, "historique": 0, "aide": 0}})
 - ('Je veux savoir combien j'ai sur mon compte', {"cats": {"solde": 1, "virement": 0, "historique": 0, "aide": 0}})
 - ('Combien j'ai sur mon compte ?', {"cats": {"solde": 1, "virement": 0, "historique": 0, "aide": 0}})
 - ('Peux-tu me donner mon solde ?', {"cats": {"solde": 1, "virement": 0, "historique": 0, "aide": 0}})
 - ('Montre-moi le montant disponible', {"cats": {"solde": 1, "virement": 0, "historique": 0, "aide": 0}})
 - ('Je voudrais consulter mon solde', {"cats": {"solde": 1, "virement": 0, "historique": 0, "aide": 0}})
 - ('Quel est mon solde actuel ?', {"cats": {"solde": 1, "virement": 0, "historique": 0, "aide": 0}})
 - ('Affiche le solde de mon compte s'il te plaît', {"cats": {"solde": 1, "virement": 0, "historique": 0, "aide": 0}})
- Sidebar:** Includes icons for file operations (New, Open, Save, etc.) and a search bar.
- Bottom Bar:** Variables, Terminal, Python 3, and a status bar showing 21:19.

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**

The screenshot shows a web-based chatbot interface for Oraflex ChatBot. At the top, there's a navigation bar with links for 'Mon solde', 'Virement', 'Historique', and 'Aide'. Below the navigation, a search bar contains the placeholder 'Posez votre question...'. A large green button on the right has a play icon and the text 'Envoyer' (Send). The main area displays three successful conversations:

- A message from the bot: "Pour transférer de l'argent vers une autre banque UEMOA, utilisez l'option 'Virement interbancaire' dans l'application." followed by a blue button labeled "je veux faire un virement".
- A message from the user: "Pour un dépôt important, contactez l'agence à l'avance afin de vérifier les procédures et limites." followed by a blue button labeled "je veux faire dépôt".
- A message from the bot: "Vous pouvez déposer de l'argent dans toutes les agences OraBank du Burkina Faso ou chez nos partenaires agréés." followed by a blue button labeled "je veux faire dépôt sur mon compte".