École nationale supérieure d'électrotechnique, d'électronique, d'informatique, d'hydraulique et des télécommunications

ENSEEIHT

Human Computer Interaction

Département Sciences du Numérique - 3A

2024-2025

Documentatioon du projet IHM Chatbot

Élèves :
Asmae Karmouchi
Safae Belahrach
Khadija Akkar

Encadrants :
Alexandre Lemort

Année Académique 2024/2025

Table des matières

Table des figures

1 Introduction

Ce document décrit le fonctionnement et la mise en œuvre de notre **Chatbot** qui permet à l'utilisateur de poser une question et d'afficher la réponse sur un tableau blanc (**Whiteboard**). Le projet est divisé en deux parties : le front-end et le back-end, tous deux développés en **JavaScript**.

Dans le cadre de ce travail, nous avons choisi de développer un projet de commandes et gestion de stocks, avec l'objectif de créer un agent conversationnel (chatbot) permettant une interaction distribuée. Ce projet exploite plusieurs fonctionnalités mises à disposition, telles que l'affichage des questions de l'utilisateur.

L'idée est de mettre en œuvre un agent capable d'interagir avec l'utilisateur tout en consultant des données liées à des produits (vêtements), comme la disponibilité d'un type de vêtements. Cela permettra une interaction fluide entre les utilisateurs et le système backend, ainsi qu'une visualisation claire des informations à travers différents canaux (interface utilisateur, tableau blanc, etc.).

2 Pré-requis

Pour exécuter ce projet, les éléments suivants sont nécessaires :

— Un éditeur de texte ou un IDE (par exemple, Visual Studio Code).

— Python3 (https ://www.python.org/downloads/) (on a travaillé avec python3 pour lancer les agents)

— Ingescape Circle version 4 (voir source d'installation en bibliographie)

— Whiteboard (voir source d'installation en bibliographie)

3 Composants du Projet

Ingescape :
Ingescape est utilisé pour assurer la communication entre les différents composants du système et garantir une interaction fluide entre l'interface utilisateur et le backend.

L'agent *UserInterface* :
L'interface utilisateur est une application web permettant aux clients de passer des commandes. Elle est conçue en JavaScript et intègre des fonctionnalités interactives pour une meilleure expérience utilisateur avec un formulaire pour les saisies utilisateur et des visualisations sous forme de listes déroulantes pour afficher les choix de couleur, catégorie, type et taille.

L'agent *Backend* : Le backend est responsable de la gestion des requêtes, du traitement des commandes et de l'interaction avec la base de données. Il assure également la validation des entrées et la cohérence des données.

Base de données (Stock) : La base de données a été utilisée pour stocker des informations sur les produits (par exemple, les jupes, pull, chemises disponibles). On a ainsi définit des tables avec des champs comme l'identifiant unique, le type de produit, la quantité disponible, et la catégorie.

L'agent *Tableau Blanc* (*Whiteboard*) : Le Whiteboard nous a permis d'afficher de manière visuelle les informations (JSON de commande effectuée) sur sa partie de conversation texte.

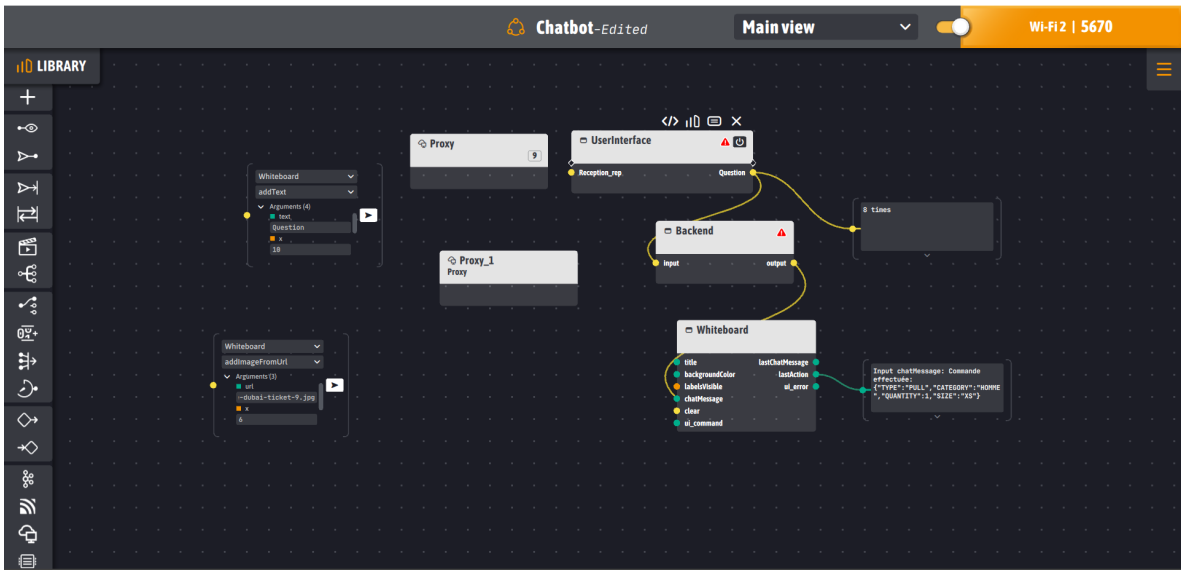

FIGURE 1 – Modélisation Ingescape du projet

4 Mise en œuvre

4.1 Front-end - L'agent *UserInterface*

Le front-end est développé en JavaScript avec l'utilisation de **HTML** et **CSS** pour l'interface utilisateur. L'utilisateur peut à l'aide d'un formulaire choisir le type, la catégorie, la couleur du vetement qu'il veut acheter, il clique à la suite sur un bouton pour envoyer sa commande à l'agent Chatbot, et la réponse est affichée sur le tableau blanc.

4.1.1 Structure du Front-end

- ***HTML*** : Structure de base de la page web.
- ***CSS*** : Style de la page pour une meilleure expérience utilisateur.
- ***JavaScript*** : Gestion des interactions utilisateur et communication avec le back-end via WebSocket.

FIGURE 2 – Modélisation Ingescape de l'agent UserInterface

Notre front-end développé interagit avec un serveur via un agent intelligent **(IGS)**, utilise des technologies JavaScript et HTML pour créer un formulaire interactif permettant à l'utilisateur de soumettre une commande.

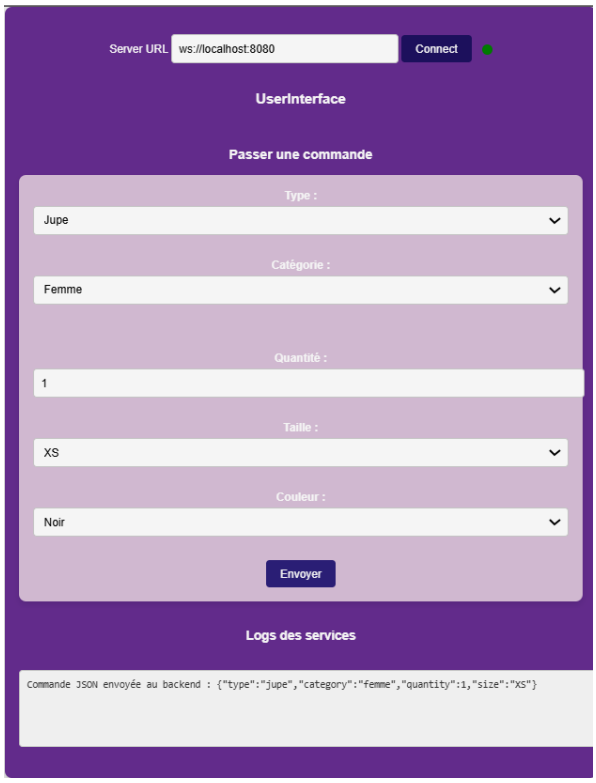

FIGURE 3 – Le formulaire interactif du front-end

En premier, on établit une connexion **WebSocket** avec le serveur (via **'IGS.netSetServerURL'**) et initie notre agent nommé **"UserInterface"**.

Lorsque l'utilisateur soumet un formulaire avec des informations sur un produit, un objet JSON représentant la commande est construit à partir des valeurs du formulaire. Ce dernier est ensuite envoyé au serveur en utilisant **'IGS.outputSetString'**.

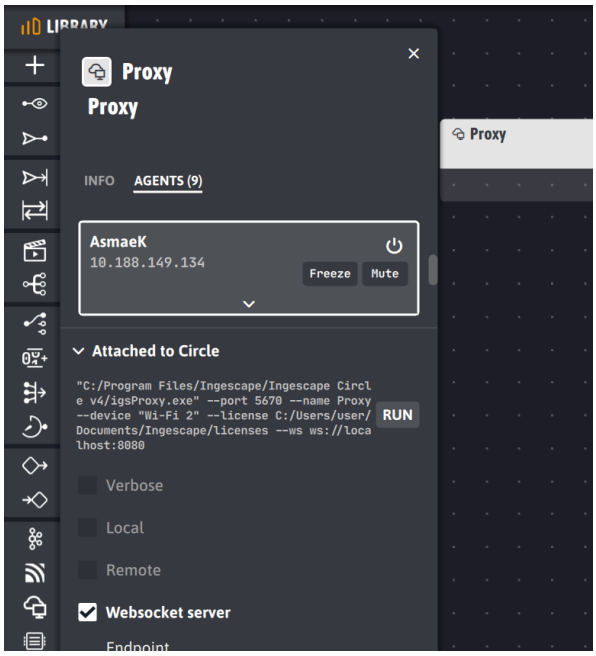

FIGURE 4 – Le configuration du proxy du front-end

En outre, on gère l'affichage en temps réel des informations sur un tableau blanc via l'agent **Whiteboard**. Si une commande est envoyée, un texte est ajouté au tableau blanc avec des paramètres définis tels que la position, la taille de la police et la couleur. Le log de ces échanges est également affiché dans une zone de texte spécifique pour la visualisation en temps réel des actions effectuées sur le formulaire.

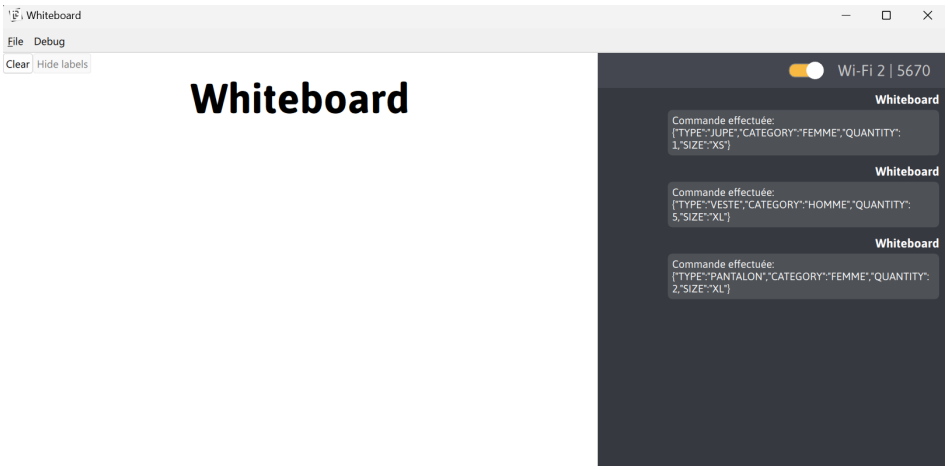

FIGURE 5 – L'affichage des messages dans le Whiteboard

Enfin, on gère aussi les erreurs qui pourraient survenir lors de l'envoi des données ou de l'affichage sur le tableau blanc, en les capturant et en les affichant dans la console pour faciliter le débogage.

Ainsi, on garantit une interaction fluide et dynamique entre l'utilisateur, le serveur et l'interface graphique.

4.2 Back-end - L'agent *Chatbot*

Le back-end Chatbot est également développé en **JavaScript**. Il gère les requêtes de l'utilisateur, interagit avec **UserInterface** pour obtenir une demande, interroge la base de donnée et envoie une réponse au front-end **UserInterface**.

FIGURE 6 – Modélisation Ingescape de l'agent Chatbot

4.2.1 Structure du Back-end

— ***JavaScript(index.js)*** : Gestion des interactions utilisateur et communication avec le front-end via WebSocket.

— ***Chatbot*** : Module qui traite la question de l'utilisateur et génère une réponse.

Le **backend - chatbot** est implémenté en Javascript et interagit avec un serveur via **WebSocket**. Il gère les entrées utilisateur, traite les données reçues et renvoie un résultat formaté en sortie.

Lorsqu'un utilisateur saisit une commande, la fonction `InputInputCallback` est déclenchée pour vérifier et transformer les données avant de les envoyer dans l'output. Un système de gestion d'état empêche le traitement simultané de plusieurs requêtes (`isProcessing`).

La connexion au serveur est surveillée dynamiquement grâce à `isConnectedToServerChanged`, qui met à jour l'interface utilisateur en fonction de l'état du **WebSocket**.

Le backend utilise l'API **IGS** pour créer et observer les entrées/sorties (`IGS.inputCreate`, `IGS.outputCreate`), et démarre l'agent via `IGS.start()`. En complément, des éléments HTML sont manipulés pour afficher l'état du serveur et configurer la connexion. Ce back-end constitue ainsi une **base fonctionnelle pour un chatbot réactif et interactif**.

4.3 Base de données - (Stock)

Notre base de données est organisée en six tables principales, qui permettent de gérer les différentes entités du système de gestion des commandes et du stock de vêtements.

- **categories** : Contient les différentes catégories de vêtements (ex. Homme, Femme, Enfant).
- **clients** : Stocke les informations des clients, y compris leur identifiant, nom et coordonnées.
- **commandes** : Regroupe toutes les commandes passées, associées aux clients et aux articles commandés.
- **stock** : Gère les quantités disponibles pour chaque type de vêtement.
- **tailles** : Définit les tailles disponibles pour les vêtements (S, M, L, XL, etc.).
- **types_vetements** : Liste les types de vêtements (pantalon, chemise, robe, etc.).

Cette structuration assure une gestion efficace et optimisée des stocks, des commandes et des interactions entre les clients et le système.

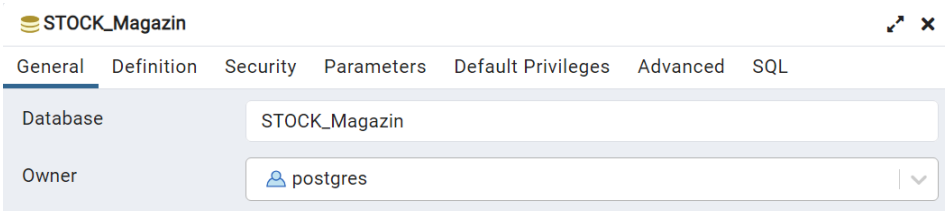

FIGURE 7 – La base de donnée - STOCK

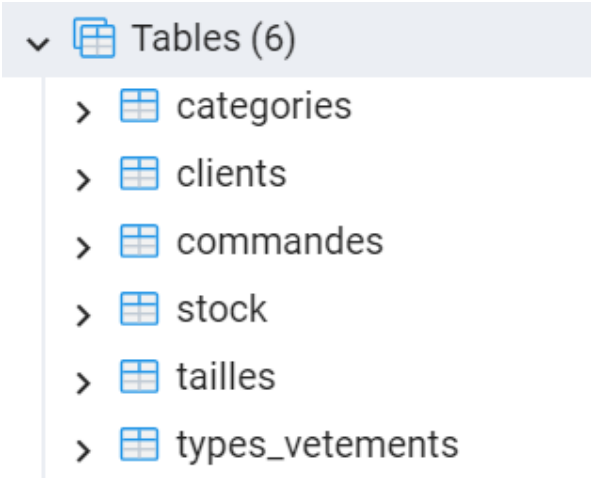

FIGURE 8 – La structure de la base de donnée

5 Résultats

Le projet fonctionne globalement comme prévu. L'utilisateur peut poser une question via l'interface du chatbot, et ceci est affichée sur le tableau blanc en temps réel grâce à l'utilisation de **WebSocket**. Cette interaction fluide permet une communication efficace et interactive.

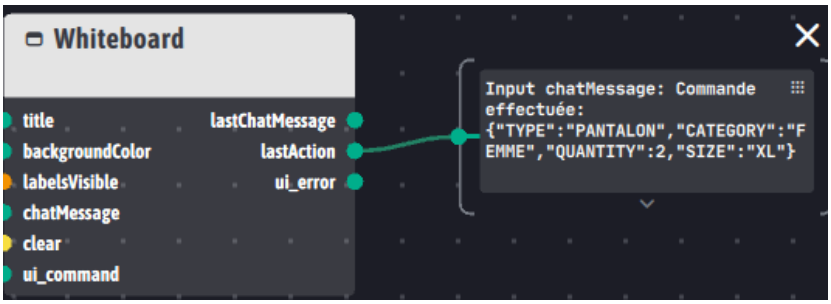

FIGURE 9 – Le résultat d'une commande

Fonctionnalités implémentées :

1. **Affichage des réponses textuelles** : Le chatbot traite la requête et affiche la réponse sur le tableau blanc.
2. **Mise à jour en temps réel** : Grâce à WebSocket, les réponses sont instantanément visibles pour tous les utilisateurs connectés.
3. **Interface réactive et interactive** : Les utilisateurs peuvent interagir avec le tableau blanc pour annoter ou modifier les informations affichées.

Limitations et axes d'amélioration : Nous avions initialement prévu d'intégrer les fonctionnalités suivantes :

- Affichage des images du stock en complément des réponses textuelles. - Indication de la disponibilité des articles via une réponse automatisée du backend confirmant ou infirmant l'existence de l'article demandé.

Cependant, ces fonctionnalités n'ont pas pu être mises en place dans cette version du projet en raison de : Contraintes de temps qui ont limité le développement et les tests approfondis. Première manipulation de la plateforme Ingescape et des agents comme Whiteboard, nécessitant une prise en main plus approfondie avant une intégration complète.

Ces éléments constituent des pistes d'amélioration pour les futures versions du projet. Une meilleure maîtrise de la plateforme et une gestion optimisée du temps de développement permettront d'intégrer ces fonctionnalités et d'améliorer l'expérience utilisateur.

6 Mode d'emploi

1. Installer les dépendances :

```
python3 -m pip install ingescape
```

2. Exécuter (Run) l'interface utilisateur (User Interface) avec la commande :

```
python3 main.py agentName device port
```

Par exemple :

```
python3 -m http.server 8600
```

3. Exécuter (Run) le backend (Chatbot) avec la commande :

```
python3 main.py agentName device port
```

Par exemple :

```
python3 -m http.server 8500
```

4. Ouvrir l'interface utilisateur (Front-end) dans votre navigateur : `http://localhost:8600/`
5. Ouvrir l'interface Back-end pour vérifier le bon démarrage du backend : `http://localhost:8500/`
6. Remplir le formulaire avec la commande voulue.
7. Appuyer sur « Envoyer » pour obtenir la réponse du Chatbot.
8. La réponse sera affichée en temps réel sur le tableau blanc (*Whiteboard*).

7 Conclusion

Ce projet a permis d'explorer et de mettre en œuvre une interaction avancée entre un **Chatbot** et une interface utilisateur intégrant un **tableau blanc interactif (Whiteboard)**. En combinant différentes technologies, nous avons conçu un système capable de traiter les requêtes des utilisateurs en temps réel et de restituer les réponses de manière visuelle et intuitive.

L'objectif principal était de développer un agent conversationnel capable de **gérer des commandes et un stock de produits (vêtements)**, tout en affichant les réponses de manière interactive sur un **Whiteboard collaboratif**. Ce projet nous a permis de travailler sur plusieurs aspects clés du développement logiciel, notamment :

— **La communication en temps réel** via WebSocket, garantissant une interaction fluide entre les utilisateurs et le chatbot.

— **L'intégration de bases de données** pour stocker et récupérer des informations sur les produits.

— **L'affichage interactif** des réponses sur le Whiteboard, améliorant l'expérience utilisateur en offrant une interface visuelle dynamique.

Références

[1] Whiteboard. Disponible en ligne : `https://ingescape.com/n7/Whiteboard/`

[2] Ingescape Cicle. Disponible en ligne : `https://ingescape.com/n7/circle/`