DJERIPERFUM
Site de vente de parfum avec Assistant intelligent de recommandation de parfums

Présenté par :

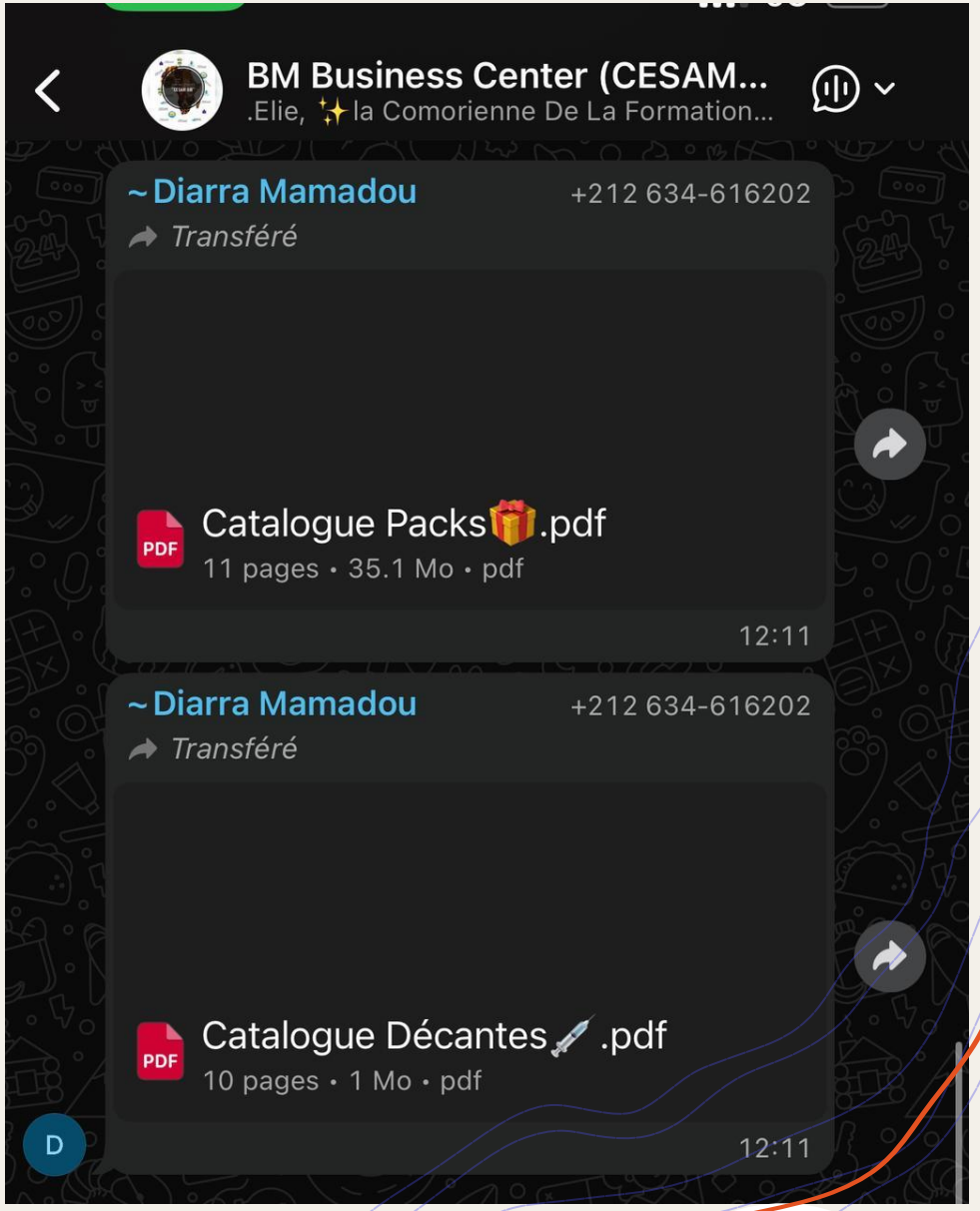
DJERI-ALASSANI OUBENOUPOU

DJERIPERFUM – Assistant intelligent de recommandation de parfums

- Dans ce projet, je présente DJERIPERFUM, un assistant intelligent de recommandation de parfums intégré dans une application web .
- L’objectif est d’aider les clients à choisir plus facilement leur parfum, à partir d’un catalogue de 10, 20 et 30 ml.
  Le projet est développé avec Python et Streamlit pour l’application web, et avec Botpress pour la partie chatbot.

Contexte et motivation

- Aujourd'hui, une grande partie des ventes de parfums se fait via internet , mais les clients sont souvent perdus face au nombre de références disponibles.
- Les vendeurs, eux, répondent sans cesse aux mêmes questions sur les prix, les familles olfactives ou le choix d'un parfum adapté à une occasion précise.
- J'ai moi-même observé un vendeur, dans un groupe de discussion, qui envoyait simplement un catalogue des parfums qu'il proposait et s est retrouver à répondre aux mêmes questions pour chaque nouveau client potentiel.
- À partir de ce constat, j'ai voulu créer un assistant digital qui présente ce catalogue de manière claire et interactive, propose des recommandations personnalisées et automatise une partie de ces réponses répétitives.
- Mon site web reprend toute la gamme de parfums de ce vendeur et l'organise dans une interface plus moderne et plus facile à utiliser.



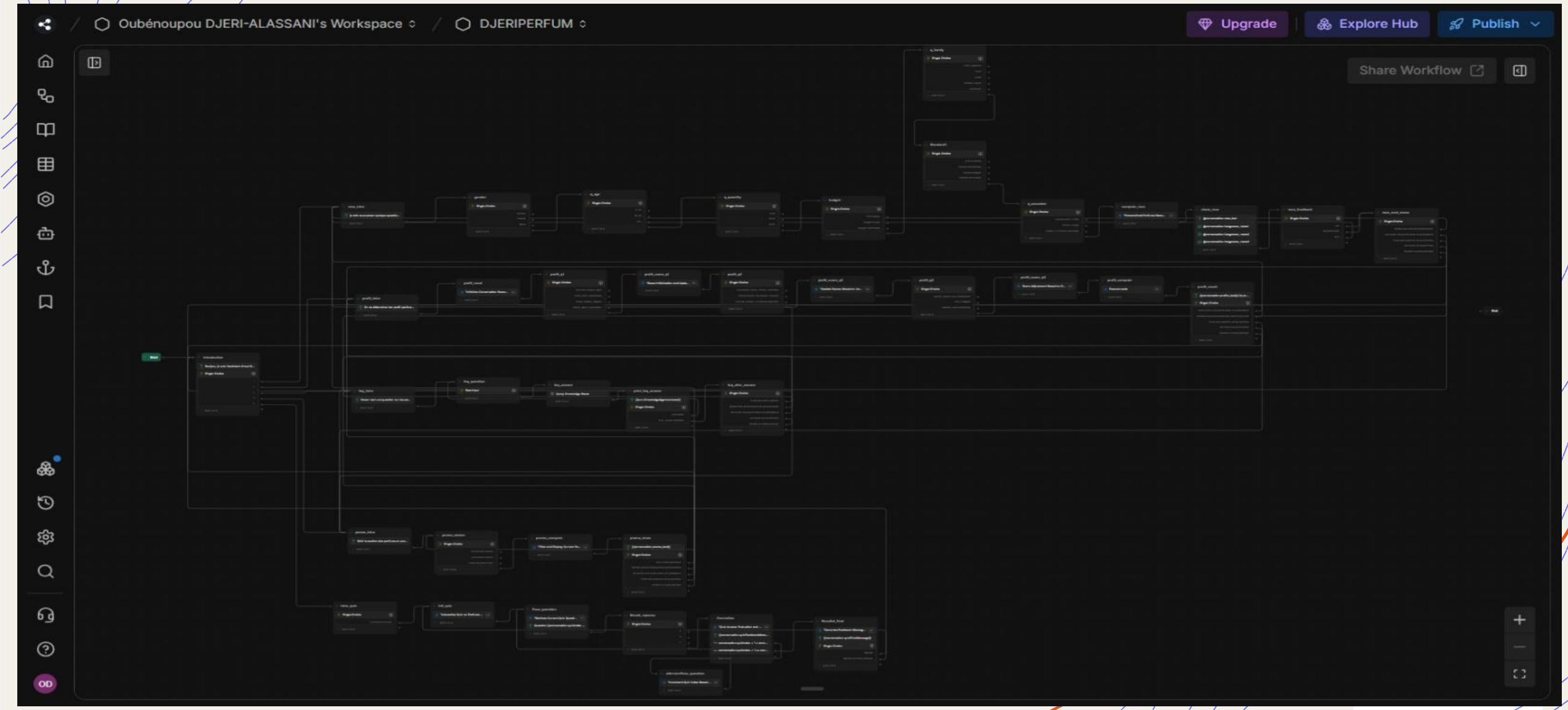
Objectifs du projet

- Le premier objectif est de développer un chatbot avec Botpress qui discute avec l'utilisateur, pose quelques questions simples, puis propose des parfums adaptés à son profil.
- Le deuxième objectif du projet est de concevoir une application Streamlit capable d'afficher l'ensemble du catalogue de parfums, ainsi qu'une fiche détaillée pour chaque référence.
- Enfin, le troisième objectif est d'intégrer ces deux éléments, de façon ergonomique et fonctionnel . Puis enfin de déployer notre site web .

Stack technique utilisée

- L'application web est développée en Python en utilisant le framework Streamlit, qui permet de créer rapidement des interfaces interactives.
- Les données sont stockées dans :
  - des fichiers CSV et TXT, qui décrivent les caractéristiques et les prix de 104 parfums.
  - les images des flacons du catalogue de vente.
- Le chatbot est construit avec Botpress, qui permet de définir des flows de conversation, des nœuds de décision, et une logique de recommandation adaptée à ce projet.

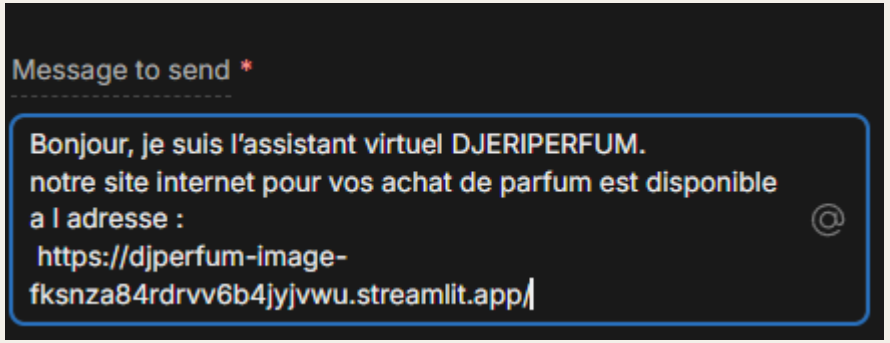
Vue globale de l'Architecture du bot dans Botpress



Architecture du bot dans Botpress

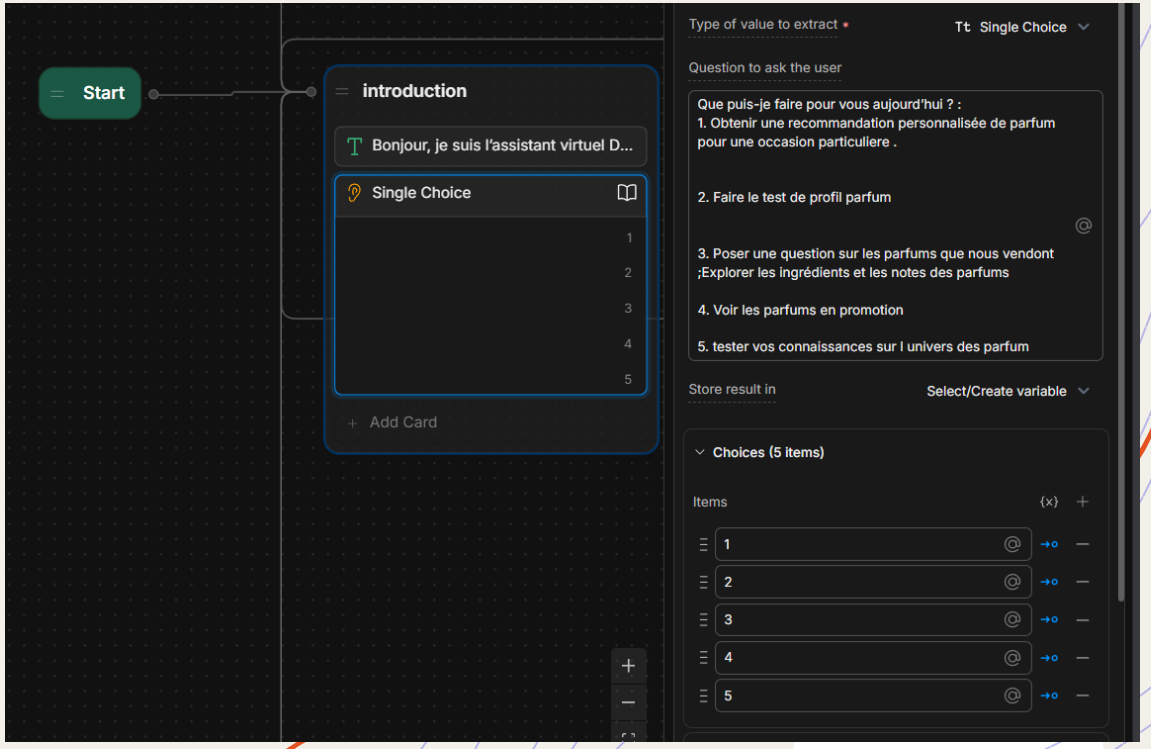
- Dans Botpress, j'ai structuré l'assistant DJERIPERFUM autour de plusieurs flows reliés entre eux.
- Le bot commence toujours par un flow d'accueil, qui présente les fonctionnalités principales et propose un menu : obtenir une recommandation, définir son profil senteur, poser une question libre ,ou voir les promotions.
Chaque fonctionnalité correspond ensuite à un flow dédié, avec ses propres nœuds, ses variables, et parfois du code JavaScript pour adapter la réponse aux préférences de l'utilisateur.

Flow principal et scénarios de conversation



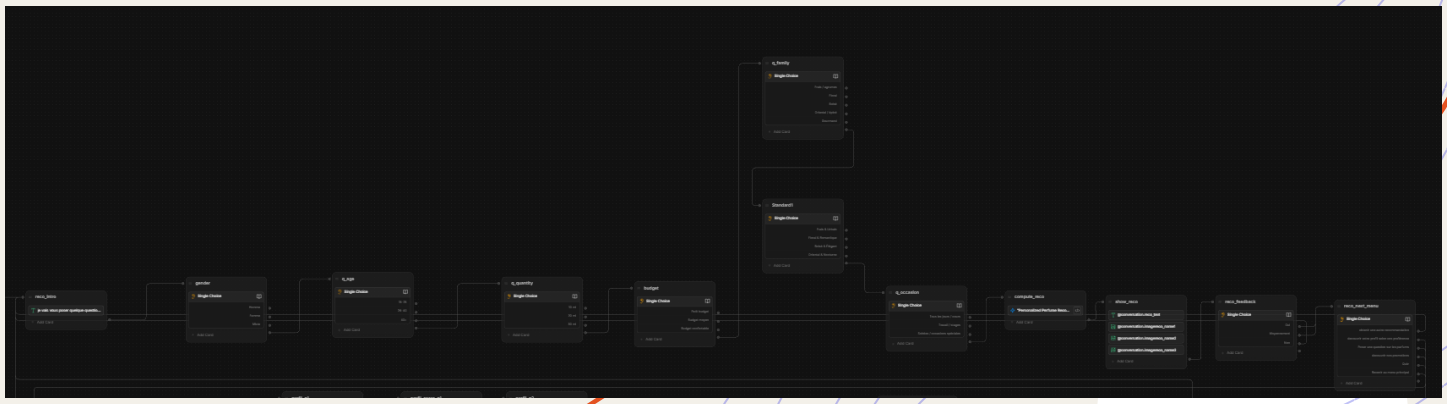
- Le flow principal sert de point d'entrée au chatbot. Il détecte le premier message de l'utilisateur et lui envoie un message de bienvenue.
  Le bot propose alors 5 scénarios :
  – une recommandation personnalisée de parfum ;
  – définir son profil senteur.
  –un mode "FAQ" où l'utilisateur peut poser ses questions en langage naturel.
  – l'affichage des promotions du moment ;
  - tester vos connaissances sur l univers des parfum

En fonction du choix de l'utilisateur, Botpress déclenche le flow approprié, ce qui permet d'avoir une conversation structurée tout en restant simple à utiliser.



Flow “Recommandation personnalisée”

Le premier flux important du chatbot est le flux de recommandation de parfum.
Dans ce flux, le bot joue le rôle de conseiller personnel. L’objectif est de proposer une sélection de parfums adaptée au profil de l’utilisateur.
Pour cela, le flux commence par une série de questions structurées.
Nous demandons successivement :
– le genre du client
– la famille olfactive qu’il préfère
– son budget
– l’occasion pour laquelle il cherche un parfum
– la quantité souhaitée
Chaque réponse est enregistrée dans des variables du conversation-state, ce qui nous permet de garder une trace cohérente du profil parfum recherché.
Une fois toutes les réponses collectées, nous utilisons un code JavaScript dans un nœud “Execute code”.
Ce code fait correspondre les labels textuels des réponses (comme “Homme”, “Frais / agrumes”, “Petit budget”, etc.) à des codes internes normalisés — par exemple homme, frais, 20ml, quotidien.
Ensuite, une logique de filtrage s’exécute : elle va parcourir le catalogue JSON chargé dans Botpress, et sélectionner uniquement les parfums correspondant aux critères.
Le bot retourne alors une liste de recommandations personnalisées, généralement 3 à 5 parfums, chacun présenté avec son nom, sa famille, et un commentaire concis.
Ce flux montre comment Botpress peut combiner des questions utilisateur, des variables de contexte et un traitement algorithmique pour produire une vraie recommandation intelligente, alignée avec notre boutique Streamlit.



Flow “test de profil parfum”

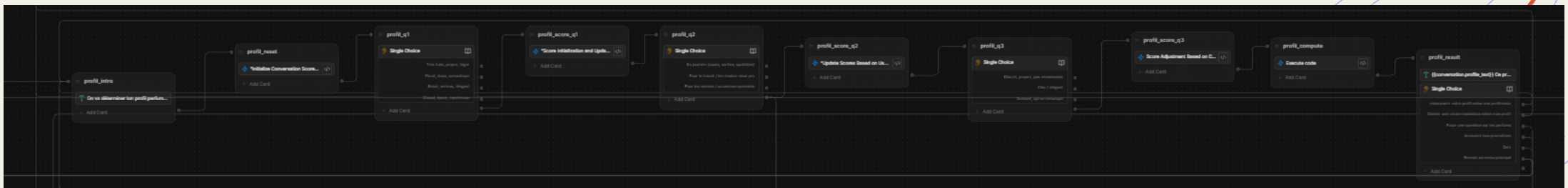
- Le deuxième flux est le flux de “Profil parfum”.
  L’objectif ici est de réaliser une sorte de mini-test pour découvrir le style olfactif dominant de l’utilisateur.

- Le flux commence par une série de questions ciblées, par exemple :
  – préfère-t-il des parfums légers ou intenses ?
  – aime-t-il les notes sucrées, boisées, florales ?
  – cherche-t-il un parfum pour tous les jours ou seulement pour les sorties ?
  – privilégie-t-il un effet discret, ou un parfum qui laisse une forte empreinte ?

- Chaque réponse est enregistrée dans une variable du state, de manière structurée.
  À la fin du questionnaire, un script JavaScript va analyser les choix de l’utilisateur.
  Par exemple, si l’utilisateur choisit souvent des options orientées “chaud”, “opulent”, “soirée”, le bot peut lui attribuer un profil comme Oriental Ambré.
  S’il préfère la fraîcheur, les agrumes et les senteurs propres, le résultat peut être Frais / Hespéridé.
  On peut aussi détecter un profil floral, boisé, gourmand, ou un profil mixte.

- Après cette analyse, le bot présente à l’utilisateur son “profil testeur”, accompagné d’une petite description :
  – ses notes dominantes,
  – les types de parfums qui lui correspondent,
  – et parfois même une mini-liste de parfums emblématiques de ce style.

- Ce flux a une valeur marketing importante, car il crée une expérience personnalisée et engageante, un peu comme un test de personnalité appliqué au parfum.



Flow “Poser une question sur les parfums que nous vendont ;Explorer les ingrédients et les notes des parfums”

+ Le troisième flux du bot est dédié aux questions libres posées par l’utilisateur.
  C’est un flux essentiel, car il transforme le bot en véritable assistant capable de répondre à des questions variées.

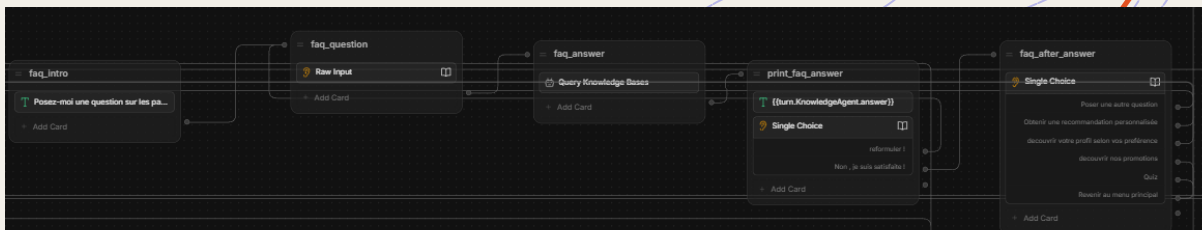
+ Pour y arriver, j’ai construit une base de connaissances composée de cinq tables distinctes :

1. Une table catalogue : prix, quantités, catégories, description.
2. Une table de compositions : les notes olfactives de chaque parfum.
3. Une table sur la gestion de la boutique : délai de préparation, modalités de commande, livraison.
4. Une table encyclopédique : définitions des notes comme patchouli, ambre, vanille, musc, etc.
5. Une table informative générale : questions fréquentes, vocabulaire du parfum, conseils d’utilisation.

+ Lorsque l’utilisateur pose une question, Botpress déclenche une carte “Query Knowledge Base”.
  Le bot cherche la meilleure correspondance dans les cinq bases de données en utilisant la recherche hybride (semantic + keyword).
  Une réponse pertinente est ensuite renvoyée automatiquement au client.

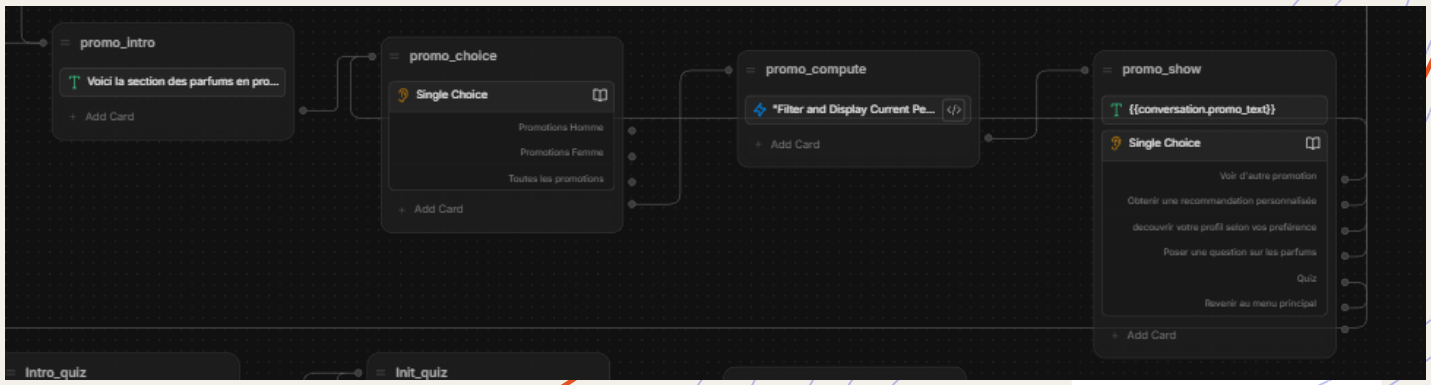
+ Ce flux permet au bot de :
  – expliquer ce qu’est un accord olfactif
  – donner le prix d’un parfum précis
  – dire combien de temps dure la préparation d’une commande
  – répondre à des questions sur les notes, les familles olfactives ou les usages

+ Ainsi, le bot peut gérer une large partie du support client sans aucune intervention humaine.



Flow “Voir les parfums en promotion”

- Le quatrième flux est celui des promotions.
  C’est un flux simple mais très utile pour orienter le client vers les meilleures offres du moment.
- Le bot commence par demander à l’utilisateur s’il cherche :
  – une promotion pour Homme
  – une promotion pour Femme
  – ou une promotion Mixte / Tout confondu
- Selon la réponse, Botpress déclenche un script JavaScript qui va filtrer les promotions stockées dans une liste interne — exactement les mêmes promotions que celles utilisées dans l’application Streamlit.
  Ce code va donc récupérer uniquement les parfums en promotion pour la catégorie choisie, et renvoyer à l’utilisateur une liste formatée, avec :
  – le nom du parfum
  – la quantité en ml
  – l’ancien prix barré
  – le nouveau prix
  – une petite phrase marketing
- Ce flux est très utile car il automatise entièrement la présentation des promotions sans nécessiter une interface manuelle.
  C’est l’équivalent du catalogue promotionnel de Streamlit, mais accessible directement via le chat.



Flow “tester vos connaissances sur l univers des parfum”

- Le dernier flux majeur est le mini-jeu des 10 questions.
  Cette fonctionnalité apporte un aspect ludique et pédagogique au chatbot.

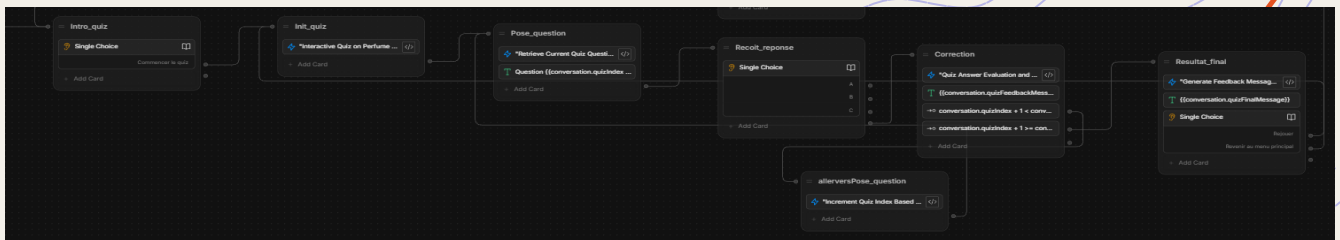
- Le fonctionnement est le suivant :
  Tout d’abord, nous avons créé un tableau JavaScript contenant les dix questions, chacune avec :
  – son identifiant
  – l’énoncé de la question
  – trois choix possibles : A, B ou C
  – la bonne réponse
  – une explication à donner après la réponse

- Un deuxième code est chargé de sélectionner la bonne question en fonction de l’index courant du joueur.
  Un troisième code compare la réponse de l’utilisateur avec la bonne réponse :
  – si la réponse est correcte, le bot le félicite
  – si la réponse est fausse, le bot donne la correction et explique pourquoi

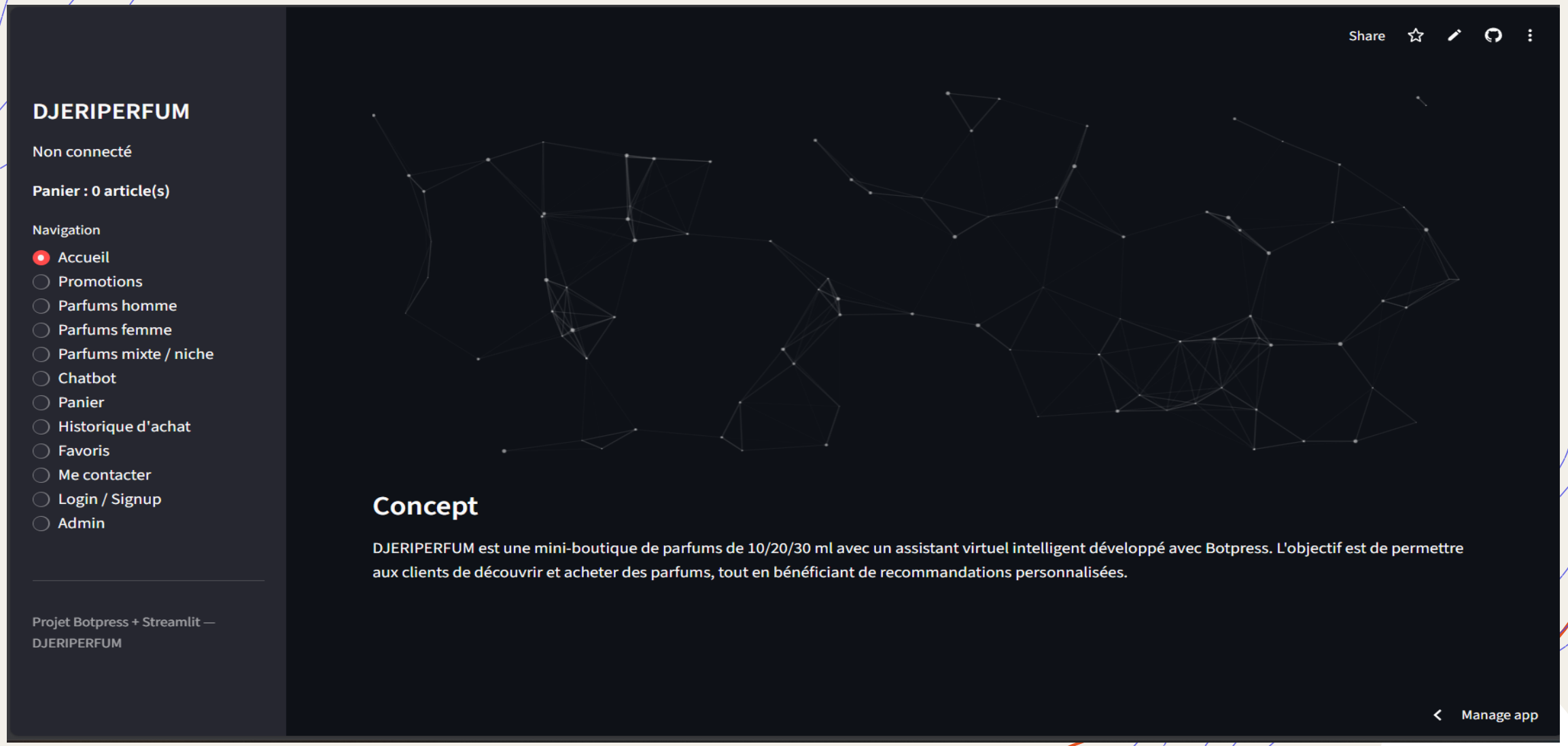
- Ensuite, une transition vérifie si l’utilisateur est arrivé à la dernière question.
  Si non, l’index est incrémenté et le bot retourne automatiquement au début du flux pour présenter la question suivante.
  Quand les 10 questions sont terminées, le bot affiche un message final avec un résumé du score ou un texte de félicitations.

- Ce flux met en avant l’utilisation combinée de :
  – variables du state
  – scripts JavaScript
  – transitions conditionnelles
  – carte Single Choice
  – gestion dynamique d’un tableau de données

- Le résultat est une activité très fluide, qui montre la puissance de Botpress pour créer des interactions complexes sans coder une application entière.



Application STREAMLIT : Architecture générale de l'application



Application STREAMLIT : Architecture générale de l'application

L'architecture de l'application est volontairement simple et facilement déployable.

En entrée, on a un fichier CSV qui contient le catalogue complet des parfums, avec les prix pour 10, 20 et 30 ml, la catégorie et l'identifiant d'image.

Les images des parfums sont stockées sur github dans un dossier, et l'application reconstruit automatiquement le chemin de chaque image à partir de l'index du CSV.

Pour les utilisateurs, je n'utilise pas de base de données lourde mais un fichier users.json. Il contient le mot de passe, le panier, les favoris et l'historique d'achat de chaque compte.

L'assistant virtuel est intégré via un composant iframe qui charge le webchat Botpress à partir d'un lien shareable.
Enfin, un module d'envoi d'e-mails via SMTP permet de traiter les messages reçus depuis la page "Me contacter".

Application STREAMLIT : Page d'accueil et expérience visuelle

La page d'accueil présente d'abord le concept de DJERIPERFUM : une boutique de décants associée à un conseiller virtuel.
J'ai ajouté une animation de fond avec la librairie JavaScript particles.js, intégrée dans Streamlit via un composant HTML.
Cette animation crée un effet de particules en mouvement qui donne un rendu plus moderne et plus professionnel à l'application, tout en restant discrète.
Depuis la page d'accueil, l'utilisateur peut rapidement comprendre l'objectif de l'application et accéder au catalogue, aux promotions ou au chatbot grâce au menu latéral.
L'idée principale, c'est que même si l'architecture est simple, l'expérience visuelle reste agréable, fluide et motivante pour le client

Application STREAMLIT : Catalogue : catégories, recherche et tri

Le catalogue est organisé en trois grandes catégories : parfums homme, parfums femme, et parfums mixtes ou niche.

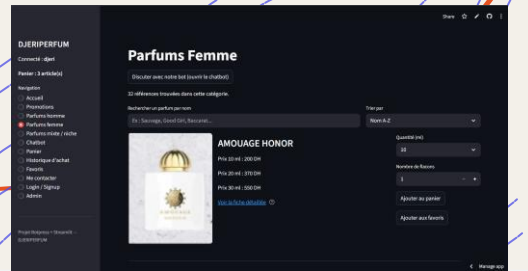
Pour chaque catégorie, l'utilisateur a accès à une liste de parfums construite à partir du fichier CSV.

Il peut rechercher un parfum par nom grâce à une barre de recherche, et trier les résultats soit par ordre alphabétique, soit par prix croissant ou décroissant pour le format voulu.

Chaque fiche résumée affiche le nom du parfum, les prix pour 10, 20 et 30 ml, ainsi que l'image associée.

Un lien "voir la fiche détaillée" permet d'ouvrir une page dédiée au parfum, en utilisant un paramètre dans l'URL pour identifier le produit.

Cette structure couvre les besoins de base d'un petit site e-commerce tout en restant simple à maintenir.



Application STREAMLIT : Fiche détaillée et composition olfactive

Quand l'utilisateur clique sur un parfum, l'application affiche une fiche détaillée construite à partir de l'identifiant du produit dans le catalogue.

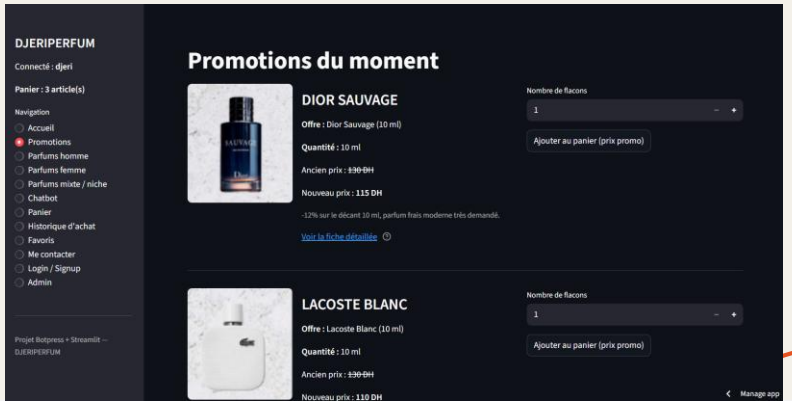
Sur cette page, on trouve une image plus grande du parfum, les prix détaillés pour les trois volumes, et des contrôles pour choisir la quantité en millilitres et le nombre de flacons à ajouter au panier.

En plus des informations commerciales, j'ai intégré un système de composition olfactive à partir d'un fichier texte.
Ce fichier contient des sections par parfum, avec la famille olfactive, les notes de tête, de cœur et de fond.
L'application lit ce fichier au démarrage et affiche la composition correspondante si elle est disponible.
Cela permet de donner au client une vision plus "expert" du parfum, comme sur les vrais sites spécialisés.

Application STREAMLIT : Promotions et intégration avec le catalogue

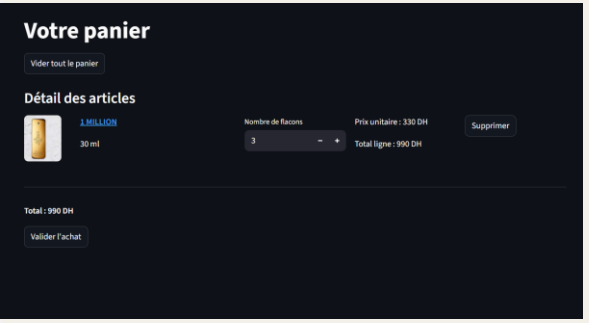
L'application propose aussi une page dédiée aux promotions du moment.

Les promotions sont définies dans un tableau Python, où chaque promotion possède un identifiant d'image, un libellé, une quantité en millilitres, l'ancien prix, le nouveau prix et une description marketing.
Pour chaque promotion, l'application retrouve automatiquement la ligne correspondante dans le catalogue afin d'afficher le vrai nom du parfum et son image officielle.
Sur l'interface, l'utilisateur voit l'ancien prix barré, le nouveau prix mis en avant, ainsi qu'un petit texte qui explique le positionnement de l'offre, par exemple "parfum frais moderne très demandé" ou "parfait pour les soirées".
Depuis cette page, un client connecté peut ajouter directement le parfum au panier au prix promotionnel, avec le nombre de flacons qu'il souhaite.
Ce système montre comment on peut combiner logique métier et données du catalogue de façon cohérente



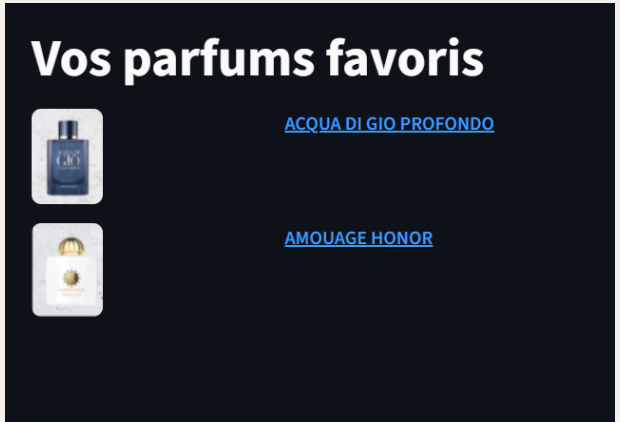
Application STREAMLIT : Panier, validation d'achat et historique

Le module panier est réservé aux utilisateurs connectés.
Lorsqu'un parfum est ajouté, l'application enregistre le nom, le prix unitaire, la quantité en millilitres et le nombre de flacons.
Dans la page "Panier", chaque ligne affiche l'image du parfum, le détail du volume, le nombre de flacons, le prix unitaire et le total de la ligne.
L'utilisateur peut modifier le nombre de flacons, supprimer une ligne ou même vider complètement le panier.
Le total général est recalculé automatiquement.
Quand il valide l'achat, l'application enregistre la commande dans l'historique de l'utilisateur avec la date, l'heure, les produits et le montant total, puis vide le panier.
La page "Historique d'achat" lui permet ensuite de consulter ses anciennes commandes avec les images des parfums et les liens vers leurs fiches



Application STREAMLIT : Favoris, compte utilisateur et persistance

En plus du panier, l'application propose un système de favoris.
Depuis les listes de parfums ou les fiches détaillées, un utilisateur connecté peut ajouter un parfum à ses favoris.
La page "Favoris" lui permet de retrouver la liste de ses parfums préférés, avec les images et les liens vers les fiches détaillées.
Côté authentification, j'ai mis en place deux onglets : un onglet Login pour se connecter avec nom d'utilisateur et mot de passe, et un onglet Signup pour créer un nouveau compte.
Toutes les informations liées au compte – mot de passe, panier, favoris, historique – sont sauvegardées dans le fichier users.json.
Cela permet à l'utilisateur de retrouver son état même après avoir fermé l'application, et c'est suffisant pour un prototype de petite boutique. »



Application STREAMLIT : Chatbot Botpress intégré à Streamlit

Un des éléments centraux du projet est l'intégration du chatbot DJERIPERFUM développé avec Botpress.
Le bot est accessible depuis plusieurs pages via un bouton qui redirige vers la page "Chatbot".
Sur cette page, j'intègre directement le webchat Botpress grâce à une iframe qui pointe vers une configuration hébergée dans le cloud Botpress.
Le rôle du bot est de conseiller le client sur le choix de parfum, en fonction de son genre, de l'occasion, de son budget, de la saison ou de l'intensité qu'il recherche.
Le chatbot peut par exemple proposer des idées pour un parfum de cours, un parfum de soirée ou un cadeau, et renvoyer les noms de parfums qui se trouvent réellement dans mon catalogue.
Cette intégration montre comment on peut connecter une interface low-code de type Botpress à une application Python personnalisée.

Application STREAMLIT :Espace administrateur et conclusion

Enfin, l'application contient un espace "Admin" réservé à certains comptes.
Dans cette section, l'administrateur peut consulter toutes les commandes passées par les utilisateurs.
Les données sont présentées sous forme de tableau : nom du client, parfum, quantité, nombre de flacons, prix unitaire, total de la ligne et total de la commande.
L'administrateur peut exporter ces informations en CSV pour faire des analyses ou un suivi des ventes, et une vue de résumé par client affiche par exemple le montant maximal de commande.
Pour conclure, DJERIPERFUM combine plusieurs aspects : une interface e-commerce simple, une gestion d'utilisateurs, un système de promotions, un chatbot de recommandation et un back-office admin.
Les perspectives d'amélioration possibles incluent l'ajout d'une vraie base de données, d'un système de paiement en ligne, et l'enrichissement du bot avec encore plus de connaissances sur les parfums

Merci!