Rapport projet JAVA

Amina Addi, Anaïs Assogane, Yasmine Gaoui, Kilian Meddas

Amina Addi, Anaïs Assogane, Yasmine Gaoui, Kilian Meddas

I. Contexte

MiamMia se distingue par son cadre idyllique et son atmosphère conviviale, qui lui ont permis de se forger une solide réputation. La localisation en bord de mer et la qualité de sa cuisine contribuent à faire de ce restaurant un lieu incontournable pour les amateurs de gastronomie méditerranéenne.

2. Une popularité en croissance

Face à son succès, le restaurant connaît un afflux constant de clients. Cette popularité, engendre également de nouveaux défis :

- Afflux massif de clients : L'augmentation de la fréquentation met sous pression la gestion quotidienne des réservations et des commandes.

- Erreurs et oublis : Les réservations oubliées et les erreurs dans les commandes se multiplient en période de forte affluence, générant du stress parmi le personnel.

3. Les enjeux de la gestion opérationnelle

Les problèmes rencontrés par MiamMia relèvent principalement de l'organisation et de la gestion :

- Gestion des réservations : La difficulté à coordonner les réservations en temps réel, à vérifier la disponibilité et à valider les informations (nombre de personnes, heure, etc.) est un point sensible.

- Prise en charge des commandes : Les erreurs dans la prise des commandes et le manque de récapitulatifs précis peuvent impacter la satisfaction des clients et l'efficacité du service.

- Archivage des données : Il est également crucial de distinguer entre les réservations à venir et les anciennes réservations, ce qui nécessite une bonne organisation des fichiers (current.json et archive.json).

4. La solution proposée : Un chatbot intelligent

Pour répondre à ces défis, la direction de MiamMia a décidé de moderniser son service grâce au développement d'un chatbot en Java. Ce chatbot doit permettre de :

- Simplifier la réservation : En offrant une interface conversationnelle via des canaux comme WhatsApp ou le site web, le chatbot doit guider les clients dans le processus de réservation.

- Automatiser la commande : Il doit aussi permettre de passer et de gérer les commandes de manière efficace, en intégrant des fonctions de récapitulatif et de calcul de l'addition.

- Optimiser la gestion des données : Le chatbot devra enregistrer et archiver les informations relatives aux réservations et aux commandes, facilitant ainsi le suivi et l'analyse des activités du restaurant.

5. Enjeux technologiques et méthodologiques

Pour ce projet nous :

- Utiliseront de la Programmation Orientée Objet (POO) en Java : Pour structurer le code et favoriser la maintenabilité et l'évolution de l'application.

- Intégreront la technologie pour le traitement du langage naturel (NLP) : Dans une version avancée, le chatbot pourra comprendre et interpréter des phrases plus complexes, améliorant ainsi l'interaction avec les clients.

- Feront de la gestion des données en JSON : Pour stocker les informations sur les commandes et les réservations, en assurant une lecture et une écriture fiables des fichier

II-Objectifs

L'objectif principal de ce projet est de développer un chatbot intelligent en Java qui modernise la gestion des réservations et des commandes pour le restaurant MiamMia. Ce chatbot doit non seulement simplifier l'interaction entre les clients et le restaurant, mais également améliorer l'efficacité opérationnelle en réduisant les erreurs et en optimisant l'organisation des données.

- **Interprétation et gestion des réservations**
  Le chatbot doit être capable de comprendre des phrases simples formulées par les clients pour détecter une intention de réservation. Par exemple, il doit reconnaître des demandes telles que « Je souhaite réserver une table pour 4 personnes à 20h » et proposer des créneaux disponibles en fonction des horaires et des capacités du restaurant. Cela implique de :
    - Valider les informations (nombre de personnes, heure, date) fournies par le client.
    - Gérer les erreurs (créneau indisponible, heure invalide) et offrir des alternatives adaptées.
- **Enregistrement et traitement des commandes**
  Le chatbot doit faciliter la prise de commandes en associant chaque réservation à une commande spécifique. Dans ce cadre, il devra :
    - Créer ou mettre à jour des fichiers JSON dédiés à chaque table pour enregistrer les plats et boissons commandés.
    - Calculer automatiquement le total des consommations en fonction des prix définis dans le fichier des produits.
    - Offrir un récapitulatif clair de la commande pour vérification et finalisation, assurant ainsi une communication sans erreur entre le client et le restaurant.
- **Archivage et gestion des réservations**
  Un volet important du projet consiste à organiser la gestion des réservations de manière efficace :
    - Les réservations à venir doivent être centralisées dans un fichier current.json.
    - Une fois les réservations effectuées ou après leur date, elles doivent être déplacées vers un fichier archive.json.
    - La suppression des fichiers JSON contenant les réservations passées doit également être automatisée pour garder un environnement de données propre et organisé.

- **Optimisation de l’expérience client et analyse de données**
  Au-delà des fonctionnalités de base, le chatbot peut intégrer des fonctionnalités avancées telles qu'une API de traitement du langage naturel (NLP) pour mieux comprendre les demandes des utilisateurs. De plus, le système pourra :

    - Fournir des recommandations personnalisées basées sur l’historique des commandes.

    - Générer des statistiques sur la fréquentation et les tendances des commandes, permettant ainsi au restaurant d'ajuster ses offres et sa gestion en fonction de l'évolution de la demande.

III-Fonctionnalités

1. Fonctionnalités attendues

Plusieurs fonctionnalités sont attendues de la part du Chatbot, le but étant d'améliorer la transaction dans le restaurant avec le client.

- Accueil de l'utilisateur et détection de ses besoins.
- Proposition de créneaux disponibles pour la réservation.
- Enregistrement des réservations et des commandes dans une base de données.
- Génération d'un récapitulatif des commandes et réservations.

2. Fonctionnalités actuelles

- Accueil de l'utilisateur avec un menu de choix qui lui demande ses besoins (réservation, commande, avis..)
- Enregistrer les réservations et les commandes dans une base de données.
- Générer un récapitulatif des commandes et réservations (en json pas encore intégré)
- Récupération des avis, commandes, produits et l'heure de réservation.

IV-Règle de gestion et contraintes techniques

Dans le cadre de ce projet, plusieurs règles de gestion et contraintes techniques ont été mises en place afin d'assurer le bon fonctionnement du système de réservation et de commande.

L'un des premiers choix techniques a été d'adopter un format de date standardisé pour garantir une cohérence dans la gestion des réservations. Ainsi, toutes les dates sont enregistrées sous le format "dd/MM/yyyy HH:mm", assurant une compréhension intuitive et une facilité de manipulation des données temporelles au sein du programme.

La gestion des fichiers JSON constitue un élément clé du projet. Afin de structurer les données et d'éviter toute confusion entre les réservations en cours et celles passées, nous avons mis en place un système de stockage en deux fichiers distincts. Le fichier current.json, stocké dans un dossier spécifique nommé reservation, contient l'ensemble des réservations à venir. Lorsqu'une réservation est terminée, elle est déplacée dans le fichier archive.json pour permettre un suivi historique tout en allégeant le fichier principal. De plus, un mécanisme a été prévu pour supprimer automatiquement les fichiers JSON obsolètes afin d'éviter toute surcharge inutile.

Un autre aspect essentiel du projet réside dans la gestion des menus. Pour faciliter l'enregistrement et la manipulation des commandes, nous avons utilisé un système d'abréviations permettant d'identifier rapidement chaque type de plat. Par exemple, "Pi" désigne une pizza, "Pa" une assiette de pâtes, et "Sa" une salade. Ce choix a permis d'optimiser le traitement des commandes tout en limitant les erreurs potentielles liées à la saisie manuelle.

Concernant l'architecture logicielle, le projet a été conçu selon une approche orientée objet (POO) afin d'en améliorer la lisibilité et la maintenabilité. Chaque élément clé, tel qu'une réservation ou un menu, est représenté par une classe spécifique, ce qui permet une meilleure structuration du code et une gestion plus efficace des fonctionnalités. Nous avons également veillé à ce que le main du programme reste le plus simple possible, se limitant au strict nécessaire pour assurer un parcours utilisateur fluide et intuitif. Par ailleurs, l'intégration de classes utilitaires a permis de regrouper les fonctionnalités génériques, telles que la lecture et l'écriture des fichiers JSON ou encore le formatage des dates, afin d'éviter la duplication de code et de garantir une meilleure réutilisation des méthodes.

Enfin, nous accordons une importance particulière aux bonnes pratiques de développement. Nous avons testé régulièrement le programme tout au long de son élaboration, ce qui a permis d'identifier rapidement d'éventuelles erreurs et de les corriger avant qu'elles ne deviennent problématiques. L'utilisation de logs et de commentaires a également été précieuse pour le débogage, facilitant le suivi du comportement du programme et l'analyse des erreurs rencontrées. En respectant ces principes, nous avons pu obtenir une application fonctionnelle, robuste et bien structurée.

V-Modification apportées

Au cours du développement du projet, plusieurs ajustements ont été effectués afin d'améliorer son fonctionnement et son ergonomie. Ces modifications ont principalement concerné la gestion des horaires, l'organisation des menus et la structuration des fichiers JSON.

L'un des premiers ajustements a été l'ajout des horaires d'ouverture du restaurant. Cette fonctionnalité permet de vérifier que les réservations effectuées respectent les heures d'ouverture, évitant ainsi toute réservation en dehors des plages autorisées. Un bug lié à la gestion des réservations a également été corrigé. Ce bug entraînait des incohérences dans l'enregistrement et l'affichage des réservations, et sa résolution a permis d'assurer un bon fonctionnement du système.

Une autre amélioration notable concerne l'optimisation du menu des commandes. Un sous-menu a été intégré, permettant de mieux structurer les différentes options disponibles et d'offrir une navigation plus fluide aux utilisateurs. De plus, une liste complète des produits disponibles a été ajoutée pour éviter toute confusion lors de la prise de commande et garantir que seules les options proposées puissent être sélectionnées.

Enfin, des ajustements ont été apportés à la gestion des fichiers JSON pour une meilleure organisation des données. Le fichier initialement nommé "reservations.json" a été renommé "current.json", afin de mieux refléter son contenu et de différencier clairement les réservations en cours de celles archivées. De plus, des commentaires ont été ajoutés dans le code afin d'améliorer sa lisibilité et de faciliter la compréhension de la structure et des fonctionnalités implémentées.