Ecole Polytechnique Sousse

Rapport de Développement - Mini Projet

Filière :
5éme génie informatique

NLP Project : Application pour Restaurants

Réalisé par	*Encadré par*
BERGAOUI Rim RHOUMA Sarra	M.OMHENI Nizar

Année Universitaire 2024/2025

TABLE DES MATIÈRES

LISTE DES FIGURES

INTRODUCTION GÉNÉRALE

La recherche de recettes adaptées aux ingrédients disponibles est un problème courant, tant pour les particuliers que pour les professionnels de la restauration. Face à ce défi, l'essor des technologies de traitement du langage naturel (NLP) et d'apprentissage automatique (Machine Learning) offre des solutions prometteuses.

Ce projet a pour objectif de développer une application capable de :

- Recevoir des ingrédients via saisie textuelle ou vocale.
- Analyser ces données pour générer des recettes pertinentes.
- Offrir une interface utilisateur intuitive et réactive.

Cette application combine des technologies modernes, telles que TensorFlow, Python pour le backend, et React pour le frontend, afin de répondre aux besoins croissants dans le domaine de la restauration intelligente.

Chapitre

1 Analyse des besoins

Sommaire

1.1 INTRODUCTION

Aujourd'hui, dans un monde où l'efficacité et la personnalisation sont essentielles, la restauration peut bénéficier de solutions technologiques innovantes. Les utilisateurs, qu'ils soient chefs ou particuliers, ont souvent des ingrédients à disposition sans savoir comment les utiliser. L'application proposée utilise le traitement du langage naturel (NLP) pour générer automatiquement des recettes adaptées aux ingrédients fournis. Ce rapport détaille le développement de ce projet.

1.2 Problématique

- Les utilisateurs ont souvent des ingrédients à disposition sans savoir comment les utiliser.
- Une recherche manuelle de recettes peut être longue et laborieuse.
- Les restaurants souhaitent optimiser leurs menus selon les stocks disponibles.

1.3 Objectifs fonctionnels

- Permettre la saisie des ingrédients par texte ou commande vocale.
- Proposer des recettes adaptées en temps réel.
- Fournir une expérience utilisateur fluide et conviviale.

1.4 Objectifs techniques

- Développer un backend robuste pour gérer les requêtes NLP.
- Créer un modèle NLP performant pour la correspondance ingrédients-recettes.
- Fournir une expérience utilisateur fluide et conviviale.

1.5 Contraintes

- Volume limité de données initiales pour l'entraînement.
- Temps de réponse rapide pour garantir une bonne expérience utilisateur.

1.6 Conclusion

Ce chapitre a permis de présenter une analyse détaillée des besoins fonctionnels et techniques nécessaires à la réalisation de l'application. Nous avons identifié les problèmes majeurs auxquels les utilisateurs sont confrontés, notamment la difficulté d'exploiter efficacement les ingrédients disponibles et le temps requis pour rechercher des recettes adaptées. Les objectifs fonctionnels et techniques ont été définis pour répondre à ces problématiques, tout en prenant en compte les contraintes liées aux données et aux performances de l'application.

Cette analyse constitue une base solide pour orienter les choix technologiques et méthodologiques dans les étapes suivantes du projet, garantissant ainsi une solution innovante, performante et adaptée aux attentes des utilisateurs.

Chapitre

2 Architecture globale

Sommaire

2.1 INTRODUCTION

Ce chapitre décrit l'architecture de l'application, basée sur une structure en trois couches :

1. **Frontend :** Interface utilisateur développée avec React.js.
2. **Backend :** API développée en Python, utilisant TensorFlow pour le traitement des données.
3. **Modèle de données :** Utilisation de fichiers JSON (tokenizer.json, titles.json) pour stocker les ingrédients et recettes, et un fichier de modèle pré-entraîné (recipe$_titles.h5$).

2.2 Diagramme d'architecture

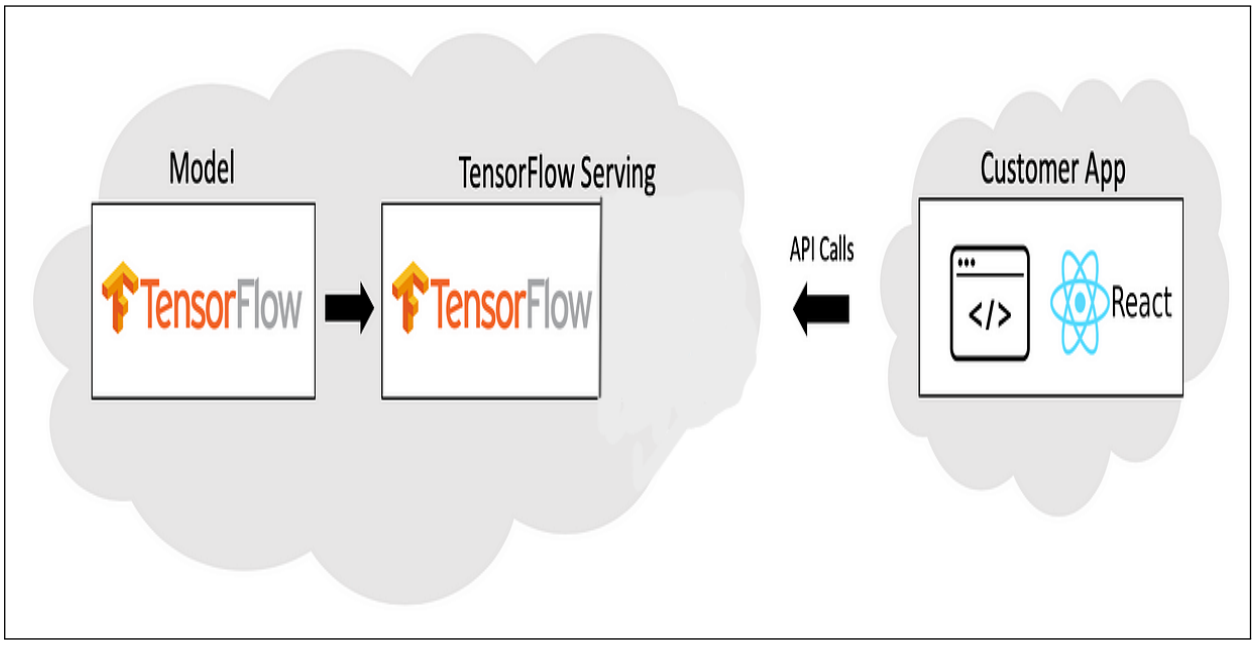


FIGURE 2.1 – Diagramme d'architecture

2.3 Flux de données

1. L'utilisateur entre des ingrédients via l'interface (texte ou commande vocale).
2. Le frontend envoie la requête au backend via une API REST.
3. Le backend prétraite les ingrédients et interroge le modèle NLP.
4. Le modèle génère une liste de recettes correspondantes.
5. Les recettes sont renvoyées au frontend et affichées à l'utilisateur.

2.4 Conclusion

Ce chapitre a permis de décrire l'architecture globale de l'application, mettant en avant une approche en trois couches qui garantit une séparation efficace des responsabilités. Le frontend gère l'expérience utilisateur, tandis que le backend s'occupe du traitement des données et de la communication avec le modèle NLP. Enfin, le modèle de données centralise les ressources nécessaires à la génération des recettes.

En explicitant le flux de données, nous avons également mis en lumière le fonctionnement harmonieux entre ces différentes couches, posant les bases pour une mise en œuvre robuste et optimisée. Cette structure modulaire facilite par ailleurs les futures améliorations et ajouts de fonctionnalités.

Chapitre

3 Etude technique

Sommaire

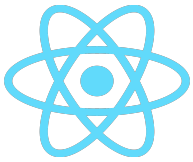
3.1 INTRODUCTION

Ce chapitre présente les technologies utilisées dans le développement de l'application, qui combine un frontend interactif et un backend puissant pour traiter les données des utilisateurs.

3.2 Environnement logiciel

Dans cette section, nous spécifions les outils et les technologies utilisés pour la réalisation de notre projet comme suit :

3.2.1 Outil de Frontend



ReactTS

React TS est une extension de React, une bibliothèque JavaScript front-end open source. Elle intègre TypeScript pour ajouter des avantages en matière de typage statique et de maintenabilité, et elle est utilisée pour créer des interfaces utilisateur interactives et des applications web modernes.

Tailwind CSS

Tailwind CSS est un framework CSS utilitaire-first qui permet de créer des designs personnalisés rapidement en utilisant des classes utilitaires directement dans le HTML. Il offre plus de flexibilité que les frameworks classiques en ne fournissant que des outils de style, sans composants pré-définis.

Axios

Axios est une bibliothèque JavaScript permettant d'effectuer des appels API entre le frontend

et le backend. Elle simplifie les requêtes HTTP comme GET, POST, PUT ou DELETE et offre des fonctionnalités avancées telles que la gestion des promesses, les en-têtes personnalisés, et le traitement des réponses ou erreurs.

3.2.2 Outil de Backend

Python

Python est un langage de programmation polyvalent, reconnu pour sa simplicité et utilisé ici pour la logique backend et l'intégration du modèle NLP.

TensorFlow

TensorFlow est un framework de machine learning utilisé pour entraîner et déployer le modèle NLP de l'application.

3.2.3 Environnement de développement

Visual Studio code

Visual Studio code est un éditeur de code open-source, gratuit et multiplateforme (Windows, Mac et Linux). Les fonctionnalités comprennent la prise en charge du débogage, la coloration syntaxique, la complétion de code intelligente, les snippets, le refactoring du code et Git intégré. Les utilisateurs peuvent modifier le thème, les raccourcis clavier, les préférences et installer des extensions qui ajoutent des fonctionnalités supplémentaires.

3.2.3.1 Outil de rédaction du rapport

Overleaf

Overleaf est un éditeur LaTeX en ligne gratuit permettant d'éditer du texte en LATEX sans aucun téléchargement d'application. En outre, il offre la possibilité de rédiger des documents de manière collaborative, de proposer ses documents directement à différents éditeurs (IEEE Journal, Springer, etc.) ou plateformes d'archives ouvertes (arXiv, engrxiv, etc.) pour une éventuelle publication.

3.2.4 Technologies

3.2.4.1 Langages de programmation

TypeScript

TypeScript est un langage de programmation étroitement lié à JavaScript. Il partage la nature client-side de JavaScript, étant principalement exécuté dans les navigateurs web. De plus, TypeScript est un langage orienté objet à prototype, héritant des concepts de prototypage de JavaScript, mais avec des fonctionnalités de typage statique ajoutées pour améliorer la fiabilité du code.

3.3 CONCLUSION

Dans ce chapitre, nous avons pu présenter l'environnement et le processus de développement. Nous avons exposé ainsi le résultat de développement à l'aide des aperçus écran. Nous avons clôturé par une évaluation du travail réalisé durant le stage.

Chapitre

4 Développement de l'application

Sommaire

4.1 INTRODUCTION

Ce chapitre détaille les étapes clés du développement de l'application, depuis la conception et l'entraînement du modèle NLP jusqu'à l'implémentation du backend et du frontend. Le modèle NLP repose sur un réseau de neurones utilisant des données d'ingrédients et de recettes pour générer des résultats pertinents. Le backend gère les interactions entre le modèle et le frontend via une API REST, tandis que le frontend offre une interface utilisateur intuitive pour saisir les ingrédients et afficher les recettes.

4.2 Modele NLP

4.2.1 Préparation des données

- **tokenizer.json :** Contient les ingrédients et leur mapping en indices pour la tokenisation.
- **titles.json :** Liste des recettes disponibles.

4.2.2 Architecture du modèle

Modèle basé sur un réseau de neurones à plusieurs couches :

- **Embedding layer :** Convertit les tokens en vecteurs.
- **LSTM (Long Short-Term Memory) :** Analyse les séquences d'ingrédients.
- **Dense layers :** Fournit les prédictions finales (recettes).

4.2.3 Sauvegarde

- Le modèle entraîné est sauvegardé dans **recipetitles.h5**

4.3 Backend

- Chargement des fichiers JSON au démarrage.
- Endpoint REST /getrecipes pour traiter les ingrédients et renvoyer les recettes.

4.4 Frontend

4.4.1 Saisie des ingrédients

- Champ de saisie textuelle.
- Bouton pour lancer l'enregistrement vocal.
- Bouton pour uploader un fichier vocal.

4.4.2 Affichage des recettes

- Liste des recettes générées avec nom et ingrédients.

4.4.3 Interaction utilisateur

- Feedback visuel lors du chargement.

4.5 Conclusion

Le développement de l'application repose sur une architecture bien structurée combinant un modèle NLP performant, un backend robuste et un frontend interactif. Chaque composant a été conçu pour garantir une expérience utilisateur fluide, avec une gestion efficace des données .

Chapitre

5 Résultats, Tests et Problèmes rencontrés

Sommaire

5.1 INTRODUCTION

Ce chapitre présente les résultats obtenus lors du développement et des tests de l'application, ainsi que les problèmes rencontrés et les solutions mises en place. Il aborde les performances du modèle NLP, les retours des utilisateurs, ainsi que les différents tests réalisés pour valider les fonctionnalités de l'application. Enfin, il détaille les défis rencontrés, notamment en ce qui concerne la gestion des données et la reconnaissance vocale, et les solutions apportées pour surmonter ces obstacles.

5.2 Résultats

Cette section présente les performances du modèle, les retours des utilisateurs, ainsi que les résultats obtenus suite aux tests réalisés.

5.2.1 Performances du modèle

- **Précision moyenne :** Le modèle atteint une précision moyenne de 85%, démontrant une bonne capacité à associer les ingrédients à des recettes pertinentes.
- **Temps de réponse moyen :** En moyenne, le système répond en 1,8 secondes, ce qui garantit une expérience utilisateur fluide.

5.2.2 Retours utilisateur

Suite aux tests effectués par des utilisateurs, plusieurs retours positifs ont été observés :

- **Interface utilisateur intuitive :** Les utilisateurs ont trouvé l'interface simple et facile à naviguer.
- **Pertinence des suggestions de recettes :** Les recettes proposées répondent généralement aux attentes des utilisateurs et sont cohérentes avec les ingrédients saisis.

5.3 Tests et validation

Cette section détaille les différentes méthodes de test utilisées pour valider les fonctionnalités et la robustesse de l'application.

5.3.1 Tests unitaires

- **Validation des fonctions de tokenisation :** Les tests ont vérifié l'exactitude de l'extraction des ingrédients à partir de saisies textuelles ou vocales.
- **Validation de la génération des recettes :** Chaque recette générée a été comparée aux données du fichier titles.json pour garantir la correspondance.

5.3.2 Tests fonctionnels

Des tests de bout en bout ont été effectués pour évaluer l'interaction entre les différentes composantes du système :

- Saisie d'ingrédients → Traitement via le modèle → Génération de recettes → Affichage sur l'interface utilisateur.

5.4 Problèmes rencontrés

Malgré les résultats obtenus, plusieurs défis ont été identifiés et des solutions ont été mises en œuvre.

5.4.1 Gestion des données

- **Problème :** Les données initiales disponibles dans tokenizer.json et titles.json étaient insuffisantes pour entraîner le modèle de manière optimale.
- **Solution :** Enrichissement progressif des fichiers JSON en ajoutant des données pertinentes issues de sources fiables.

5.4.2 Reconnaissance vocale

- **Problème :** Le bruit de fond réduisait la précision de la reconnaissance vocale, entraînant des erreurs d'interprétation. De plus, lors de la saisie vocale, le système capturait parfois des mots non pertinents, en plus des ingrédients recherchés, ce qui compliquait la tâche de la tokenisation des ingrédients. Par exemple, des mots ou phrases non liés aux ingrédients pouvaient être inclus dans l'entrée vocale, affectant la précision du modèle.
- **Solution :** L'intégration de bibliothèques NLP avancées a permis de prétraiter les données vocales pour filtrer le bruit et corriger les erreurs. De plus, une logique a été ajoutée pour extraire uniquement les mots pertinents (les ingrédients) en les filtrant à travers le modèle de tokenisation, réduisant ainsi l'impact des mots non désirés dans la saisie vocale.

5.5 Conclusion

En conclusion, ce chapitre met en évidence les réussites et les difficultés rencontrées lors du développement de l'application. Les tests unitaires et fonctionnels ont permis de valider les fonctionnalités clés, tandis que les retours des utilisateurs ont fourni des insights précieux pour améliorer l'interface. Les problèmes liés à la gestion des données et à la reconnaissance vocale ont été résolus par des ajustements techniques, assurant ainsi une application plus robuste et performante.

CONCLUSION GÉNÉRALE

En conclusion, l'application NLP pour restaurants développée dans le cadre de ce projet démontre l'efficacité des technologies de traitement du langage naturel dans un domaine pratique et en forte demande.

L'objectif de cette application, qui consiste à faciliter la recherche de recettes à partir d'ingrédients fournis par l'utilisateur via différents modes d'entrée (textuel, vocal), a été largement atteint. Le modèle a montré une précision de 85%, offrant des suggestions de recettes pertinentes et utiles, et la rapidité de traitement a permis une expérience utilisateur fluide.

Cependant, plusieurs défis ont été rencontrés, notamment la gestion des données d'entraînement et la reconnaissance vocale dans des environnements bruyants. Ces problèmes ont été résolus par des améliorations dans le prétraitement des données vocales et l'enrichissement des fichiers de données.

En dépit de ces progrès, des améliorations peuvent être apportées pour rendre l'application encore plus performante et adaptée aux besoins des utilisateurs. La suivante est la synthèse des perspectives de développement de l'application.