



Université : UIR

Département : Cybersécurité

Module : Artificial Intelligence

Rapport de projet

Morocco Smart Guide

Système intelligent de recommandation et de planification touristique

Date de remise du rapport : *12 janvier 2026*

Encadrant : Hasna El HAJI & Hamza GAMOUH

Étudiants :

- CISSE Marwane Mouawiyah
- LANKOANDE Melwine Ange Lormel Lankoulmani
- SOUNDÀ Abdelkerim Ahmat
- TILENGAR Mbaihornom Lionel
- ZONGO Nabonswendé Regis Epiphane

Table des matières

Introduction	3
I. Motivation et problématique	4
II. Définition du problème	5
III. Conception et architecture du système.....	5
IV. Base de données et intégration backend.....	6
V. Méthodologie IA (RAG & génération d'itinéraires)	7
VI. Développement du frontend	8
VII. 7. Tests et validation de l'application	10
Premier test sur Marrakech	10
Deuxième test sur Rabat.....	12
Dernier test sur le chatbot.....	13
VIII. Difficultés rencontrées et solutions apportées.....	14
Conclusion.....	16

Introduction

L'évolution rapide des technologies numériques et de l'intelligence artificielle a profondément transformé de nombreux secteurs, notamment celui du tourisme. Aujourd'hui, les voyageurs disposent d'un accès quasi illimité à des informations issues de plateformes de réservation, de moteurs de recherche, de guides en ligne et de réseaux sociaux. Toutefois, cette abondance de contenus, souvent non structurés et redondants, complique l'identification d'informations réellement pertinentes et adaptées aux besoins spécifiques de chaque visiteur. Dans ce contexte, la capacité à filtrer, organiser et personnaliser l'information devient un enjeu majeur.

Parallèlement, le Maroc s'impose comme une destination touristique de premier plan, attirant chaque année un nombre croissant de visiteurs internationaux. Cette attractivité est renforcée par l'organisation d'événements d'envergure tels que la Coupe d'Afrique des Nations, qui favorisent l'arrivée de touristes étrangers aux profils variés et disposant de contraintes de temps parfois importantes. Dès lors, la mise à disposition d'outils numériques intelligents capables d'accompagner ces visiteurs dans la découverte du pays apparaît comme une nécessité pour améliorer leur expérience globale.

Dans ce cadre, l'intelligence artificielle offre des perspectives intéressantes, en particulier à travers l'utilisation de modèles de langage et de systèmes de recherche sémantique. Ces technologies permettent d'interpréter des requêtes formulées en langage naturel et de générer des recommandations personnalisées. Cependant, leur utilisation isolée peut conduire à des résultats imprécis ou incohérents, ce qui souligne l'importance d'une intégration maîtrisée au sein d'architectures structurées et contrôlées.

Le présent rapport s'inscrit dans cette dynamique en présentant le projet *Morocco Smart Guide*, une application d'intelligence artificielle dédiée à l'assistance touristique. Ce document vise à détailler la motivation du projet, la définition du problème abordé, les choix de conception et d'architecture, ainsi que les aspects liés à l'intégration backend et frontend. Enfin, une phase de test de l'application, accompagnée de captures d'écran, permet d'illustrer le fonctionnement concret de la solution développée avant d'aboutir à une conclusion générale.

Afin de garantir la transparence, la reproductibilité et l'accessibilité du travail réalisé, l'ensemble du code source de l'application Morocco Smart Guide, incluant les composants backend, frontend et les modules d'intelligence artificielle, est mis à disposition dans un dépôt GitHub public. Ce dépôt permet d'examiner l'implémentation détaillée du système et d'accéder aux ressources nécessaires à sa réutilisation ou à son extension.

Dépôt GitHub : https://github.com/Renzo0023/Morocco_Smart_Guide.git

I. Motivation et problématique

Le choix du domaine touristique repose sur l'importance stratégique du tourisme dans le contexte marocain ainsi que sur les défis liés à l'accès et à la structuration de l'information. Le Maroc bénéficie d'un patrimoine culturel, historique et géographique particulièrement riche, attirant chaque année un nombre croissant de visiteurs internationaux. Cette attractivité est renforcée par l'organisation d'événements majeurs tels que la Coupe d'Afrique des Nations, qui favorisent l'accueil de touristes étrangers aux profils variés. Dans ce contexte, la mise à disposition d'outils numériques capables d'accompagner efficacement ces visiteurs devient un enjeu central pour améliorer l'expérience touristique.

Cependant, la préparation d'un séjour repose encore largement sur des recherches manuelles effectuées à partir de sources hétérogènes. Les plateformes existantes proposent généralement des listes de lieux ou des suggestions génériques, sans réelle prise en compte des préférences individuelles, du budget ou des contraintes temporelles. Cette situation met en évidence une problématique de personnalisation et de cohérence dans l'organisation des activités touristiques.

Par ailleurs, l'émergence des modèles de langage généralistes tels que GPT ou Gemini a démontré leur capacité à produire des recommandations touristiques en langage naturel. Toutefois, ces modèles restent limités lorsqu'ils sont utilisés de manière isolée. En effet, les réponses générées reposent principalement sur des connaissances générales et peuvent inclure des informations imprécises, non vérifiables ou incohérentes sur le plan temporel. De plus, ces modèles ne garantissent ni la structuration des résultats sous forme de planning exploitable, ni le respect strict des contraintes fournies par l'utilisateur.

La motivation de ce projet réside ainsi dans la conception d'une solution dépassant les limites d'un simple agent conversationnel basé sur un modèle de langage. L'objectif consiste à proposer un système intelligent combinant recherche sémantique sur une base de connaissances dédiée, raisonnement algorithmique pour la planification des activités et génération contrôlée de contenu. Cette approche permet d'assurer que les recommandations produites reposent sur des données structurées, qu'elles respectent les contraintes définies et qu'elles soient directement utilisables sous forme d'itinéraires organisés.

Ainsi, la problématique centrale s'articule autour de la conception d'un assistant touristique intelligent capable de fournir des recommandations fiables, personnalisées et cohérentes, tout en garantissant un niveau de contrôle et de structuration supérieur à celui offert par les modèles de langage généralistes utilisés seuls.

II. Définition du problème

La problématique abordée dans ce projet consiste à concevoir un système capable d'assister les visiteurs dans l'organisation de leur séjour touristique de manière automatique et personnalisée. Contrairement aux recommandations génériques proposées par de nombreuses plateformes, le système recherché doit être en mesure d'exploiter des données touristiques structurées afin de produire des itinéraires cohérents et directement exploitables.

Dans le contexte du tourisme marocain, cette problématique est accentuée par la diversité des destinations, des activités et des profils de voyageurs. La prise en compte simultanée de plusieurs paramètres, tels que la durée du séjour, le budget, les centres d'intérêt et les contraintes pratiques, représente un défi majeur. En effet, l'absence d'une structuration temporelle claire et d'un raisonnement explicite sur l'organisation des activités limite fortement l'utilité des recommandations générées de manière purement textuelle.

En outre, l'utilisation de modèles de langage généralistes pose des limites en matière de fiabilité et de contrôle. Ces modèles peuvent produire des réponses plausibles mais inexactes, ou proposer des activités incompatibles entre elles sur le plan temporel ou géographique. Il devient donc nécessaire de définir un cadre dans lequel la génération de contenu s'appuie sur une base de connaissances dédiée et sur des mécanismes de contrôle garantissant le respect des contraintes définies.

Ainsi, le problème central peut être formulé comme suit : comment concevoir un assistant touristique intelligent capable de sélectionner des lieux pertinents à partir de données structurées, de les organiser dans un planning temporel cohérent et d'enrichir ce planning par une génération de contenu contrôlée, tout en offrant une interaction simple et intuitive à l'utilisateur. La résolution de ce problème implique la combinaison de techniques de recherche sémantique, de planification algorithmique et de génération de langage naturel au sein d'une architecture logicielle robuste.

III. Conception et architecture du système

La conception du système repose sur une architecture modulaire visant à assurer une séparation claire entre les différentes responsabilités fonctionnelles et techniques. Cette approche permet d'intégrer des mécanismes d'intelligence artificielle avancés tout en conservant un contrôle strict sur les données manipulées et les résultats produits. Dans cette optique, l'architecture adoptée distingue explicitement l'interface utilisateur, le backend applicatif et les composants dédiés au traitement intelligent de l'information.

Dans un premier temps, l'interface utilisateur a été développée à l'aide de Streamlit afin de proposer une interaction simple et intuitive. Cette interface permet la saisie des paramètres de voyage, la visualisation d'itinéraires structurés et l'accès à un chatbot touristique conversationnel. La communication avec le backend s'effectue exclusivement via des requêtes HTTP, ce qui garantit l'indépendance entre la couche de présentation et la logique métier. Cette séparation facilite également l'évolution future de l'interface sans impact sur les traitements internes.

Ensuite, le backend applicatif repose sur le framework FastAPI, choisi pour ses performances et sa capacité à exposer des services REST clairs et typés. Les endpoints définis permettent respectivement la génération d'itinéraires personnalisés, l'interaction avec un chatbot doté de mémoire de session et la recommandation de lieux touristiques. La validation des données échangées est assurée par des modèles Pydantic, ce qui permet de garantir la cohérence des entrées et des sorties tout au long du traitement.

Par ailleurs, la couche d'intelligence artificielle constitue le cœur du système. Les paramètres de configuration sont centralisés dans un fichier `.env`, permettant de définir de manière explicite le modèle de langage utilisé, le modèle d'embeddings, les chemins de données et la clé d'accès aux services Hugging Face. Le modèle de génération retenu est `mistralai/Mistral-7B-Instruct-v0.2`, utilisé via l'API Hugging Face Inference afin de produire des réponses textuelles contrôlées. En complément, la vectorisation des données repose sur le modèle `sentence-transformers paraphrase-multilingual-MiniLM-L12-v2`, choisi pour sa légèreté et sa capacité à gérer des contenus multilingues.

Les données touristiques sont stockées sous forme de fichiers CSV et chargées dynamiquement à partir du répertoire défini par la variable d'environnement `DATA_DIR`. Ces données sont transformées en documents textuels enrichis de métadonnées, puis vectorisées avant d'être indexées dans une base vectorielle FAISS persistée localement, conformément au chemin spécifié par `FAISS_INDEX_PATH`. Ce choix permet d'assurer une recherche sémantique rapide et fiable, sans dépendre d'une infrastructure cloud externe, tout en garantissant la reproductibilité du système.

En complément, la génération des itinéraires repose sur une approche hybride combinant recherche sémantique, raisonnement algorithmique et génération contrôlée de texte. Les lieux sélectionnés via FAISS sont d'abord organisés par un algorithme de planification déterministe tenant compte de la durée des visites, du nombre de jours et des moments de la journée. Ce planning est ensuite transmis au modèle de langage uniquement pour enrichir les descriptions et améliorer la cohérence narrative, sans modifier la structure imposée. Cette conception permet de se distinguer clairement d'un simple usage de modèle de langage généraliste, en garantissant des résultats structurés, cohérents et exploitables.

Enfin, le chatbot touristique s'appuie sur une chaîne RAG conversationnelle intégrant une mémoire de session légère stockée en mémoire vive. Cette mémoire permet de conserver le contexte des échanges tout au long d'une interaction, améliorant ainsi la pertinence des réponses fournies. L'ensemble de ces composants s'inscrit dans une architecture cohérente, extensible et conforme aux exigences du projet, tout en illustrant une intégration maîtrisée des technologies d'intelligence artificielle modernes.

IV. Base de données et intégration backend

La gestion des données dans le système repose sur une approche orientée connaissances, adaptée aux besoins d'un assistant touristique intelligent. Contrairement aux architectures traditionnelles basées sur des bases de données relationnelles, le projet s'appuie sur une base vectorielle afin de permettre une recherche sémantique efficace à partir de requêtes exprimées en langage naturel. Ce choix est motivé par la nature textuelle des données manipulées et par la nécessité de fournir des recommandations pertinentes et contextualisées.

Les données touristiques sont initialement stockées sous forme de fichiers CSV, chargés dynamiquement à partir du répertoire défini par la variable d'environnement `DATA_DIR`. Chaque fichier contient des informations structurées décrivant des lieux ou activités touristiques, telles que le nom, la ville, la catégorie, le budget, la durée recommandée, le meilleur moment de visite, les descriptions et les conseils pratiques. Lors du chargement, un processus de normalisation est appliqué afin de gérer les variations de formats, de séparateurs et de types de données, garantissant ainsi la cohérence des informations exploitées par le système.

Ces données sont ensuite transformées en objets métier représentant des lieux touristiques, puis converties en documents exploitables par le moteur de recherche sémantique. Le contenu textuel de chaque document regroupe les informations descriptives essentielles, tandis que les métadonnées associées permettent de conserver des attributs structurés tels que la ville, la catégorie ou le budget. Cette séparation entre contenu vectorisé et métadonnées facilite à la fois la recherche sémantique et le filtrage contextuel des résultats.

L'indexation des documents repose sur l'utilisation d'un modèle d'embeddings multilingue défini par la variable d'environnement `EMBEDDING_MODEL_NAME`, à savoir `sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2`. Les vecteurs générés sont stockés dans une base FAISS persistée localement, conformément au chemin spécifié par `FAISS_INDEX_PATH`. Cette base vectorielle constitue la principale source de connaissances du système et permet une récupération rapide des documents les plus pertinents en fonction d'une requête donnée.

L'intégration backend est assurée par le framework FastAPI, qui centralise l'accès à la base vectorielle et expose des services REST dédiés. Lorsqu'une requête est reçue, le backend interroge l'index FAISS afin de récupérer un ensemble de documents candidats correspondant au contexte fourni. Ces documents peuvent ensuite être utilisés directement pour recommander des lieux ou servir de contexte à un mécanisme de Retrieval-Augmented Generation. L'ensemble du processus est orchestré de manière transparente par le backend, garantissant une interaction fluide entre les données, les mécanismes d'intelligence artificielle et les services exposés à l'interface utilisateur.

Ainsi, la combinaison d'une base vectorielle locale et d'une intégration backend modulaire permet d'assurer une gestion efficace des données touristiques, tout en offrant des performances adaptées aux besoins du projet. Cette approche contribue également à la fiabilité et à la reproductibilité du système, en limitant les dépendances externes et en assurant un contrôle complet sur les données et les traitements effectués.

V. Méthodologie IA (RAG & génération d'itinéraires)

La méthodologie d'intelligence artificielle adoptée dans ce projet repose sur une approche hybride combinant recherche sémantique, raisonnement algorithmique et génération contrôlée de texte. Cette démarche vise à tirer parti des capacités des modèles de langage tout en limitant leurs faiblesses, notamment en matière de cohérence et de fiabilité. Dans ce cadre, le principe du Retrieval-Augmented Generation constitue un élément central de la solution proposée.

Le mécanisme de Retrieval-Augmented Generation s'appuie sur une base de connaissances dédiée, indexée sous forme vectorielle. Lorsqu'une requête est formulée, celle-ci est d'abord transformée en vecteur à l'aide d'un modèle d'embeddings multilingue. Une recherche de similarité est ensuite effectuée au sein de la base FAISS afin d'identifier les documents les plus pertinents. Les informations récupérées servent alors de contexte contrôlé pour la génération de réponses, ce qui permet de réduire significativement le risque d'hallucinations et d'ancrer les résultats dans des données réelles et vérifiables.

Dans le cas du chatbot touristique, cette approche permet de fournir des réponses contextualisées en tenant compte à la fois de la question courante et de l'historique de la conversation. Une mémoire conversationnelle légère est utilisée afin de conserver les échanges précédents au cours d'une même session, améliorant ainsi la continuité et la pertinence des réponses fournies. Le modèle de langage intervient uniquement après la phase de récupération des documents, ce qui garantit que les informations générées reposent sur la base de connaissances du système.

En ce qui concerne la génération d'itinéraires, la méthodologie repose sur une distinction claire entre la phase de sélection des lieux et la phase d'organisation temporelle. Les lieux touristiques sont d'abord sélectionnés par recherche sémantique à partir du profil de voyage fourni, incluant la ville, la durée du séjour, le budget et les centres d'intérêt. Cette sélection permet d'identifier un ensemble de lieux candidats pertinents, issus directement de la base vectorielle.

Une fois les lieux sélectionnés, un algorithme de planification déterministe est appliqué afin d'organiser les activités sur plusieurs jours et selon des créneaux temporels précis. Cette planification tient compte de la durée estimée de chaque visite et du moment de la journée le plus approprié, permettant ainsi de produire un planning cohérent et réaliste. Ce choix méthodologique vise à garantir la faisabilité de l'itinéraire avant toute intervention du modèle de langage.

Enfin, le modèle de langage est utilisé dans une phase d'enrichissement contrôlé du planning généré. Le planning pré-calculé est transmis au modèle sous la forme d'une structure JSON strictement définie, accompagnée d'instructions explicites interdisant toute modification de la structure, des horaires ou de la liste des activités. Le rôle du modèle se limite alors à l'amélioration des descriptions, des conseils pratiques et de la cohérence narrative globale. Cette approche permet de produire des itinéraires personnalisés, structurés et directement exploitables, tout en se distinguant clairement d'une génération libre basée uniquement sur un modèle de langage généraliste.

VI. Développement du frontend

Le développement du frontend vise à fournir une interface claire, intuitive et fonctionnelle permettant d'exploiter efficacement les services d'intelligence artificielle proposés par le backend. Dans cette optique, le framework Streamlit a été retenu en raison de sa simplicité de mise en œuvre et de sa capacité à produire rapidement des interfaces interactives adaptées aux projets orientés données et intelligence artificielle.



L'interface utilisateur est structurée autour de deux fonctionnalités principales, accessibles via un système de navigation latéral. La première fonctionnalité concerne la génération d'itinéraires personnalisés, tandis que la seconde permet l'interaction avec un chatbot touristique conversationnel. Cette organisation facilite la compréhension et l'utilisation de l'application en séparant clairement les cas d'usage proposés.

Dans le cadre de la génération d'itinéraires, l'interface permet la saisie des paramètres essentiels du séjour, tels que la ville principale, la durée, le budget, les centres d'intérêt, les contraintes éventuelles et la langue souhaitée. Ces informations sont transmises au backend sous forme de requête HTTP, déclenchant le pipeline de génération d'itinéraires. Les résultats reçus sont ensuite affichés de manière structurée, sous forme de planning journalier réparti en créneaux matin, après-midi et soir. Cette présentation vise à rendre l'itinéraire immédiatement lisible et exploitable par l'utilisateur.

Par ailleurs, l'interface met en valeur les informations pratiques associées à chaque activité, notamment les horaires et les liens vers Google Maps. Cette intégration facilite la localisation des lieux recommandés et renforce l'utilité concrète de l'application dans un contexte réel de déplacement touristique. L'affichage est conçu de manière à privilégier la lisibilité et la hiérarchisation de l'information, sans surcharger l'utilisateur.

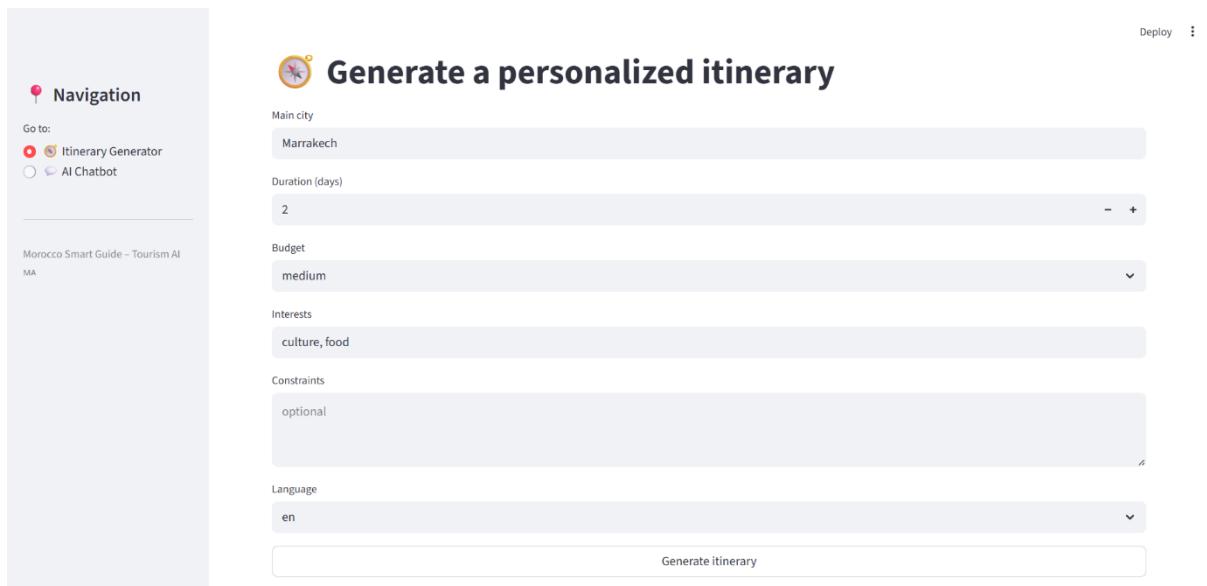
En complément, le frontend intègre un chatbot touristique reposant sur une interaction conversationnelle continue. La gestion de l'état de la session est assurée côté interface, permettant de conserver l'historique des échanges et de transmettre un identifiant de session au backend. Cette approche garantit une continuité dans la conversation et améliore la pertinence des réponses fournies par le système.

Ainsi, le développement du frontend constitue un élément clé de la solution proposée, en assurant une interface fluide entre l'utilisateur et les mécanismes d'intelligence artificielle déployés côté backend. L'utilisation de Streamlit permet de répondre efficacement aux exigences du projet tout en offrant une base solide pour des évolutions futures de l'interface.

VII. 7. Tests et validation de l'application

Afin de valider le bon fonctionnement de l'application et d'évaluer la pertinence des résultats produits, plusieurs tests ont été réalisés à partir de scénarios représentatifs d'un usage réel. Ces tests portent à la fois sur la génération d'itinéraires personnalisés et sur l'interaction avec le chatbot touristique. Les résultats obtenus sont illustrés par des captures d'écran permettant de visualiser concrètement le comportement de l'application.

Premier test sur Marrakech



The screenshot shows a user interface for generating a personalized itinerary. On the left, there's a sidebar with a navigation menu including 'Itinerary Generator' and 'AI Chatbot'. The main area has a title 'Generate a personalized itinerary' with a sun icon. It includes fields for 'Main city' (Marrakech), 'Duration (days)' (2), 'Budget' (medium), 'Interests' (culture, food), 'Constraints' (optional), and 'Language' (en). A 'Generate itinerary' button is at the bottom.

Le premier test concerne la génération d'un itinéraire touristique pour la ville de Marrakech. Dans ce scénario, les paramètres fournis correspondent à un séjour de deux jours, avec un budget moyen et des centres d'intérêt orientés vers la culture et la gastronomie. La langue de génération a été définie en anglais afin de simuler un cas d'utilisation par un visiteur étranger. La capture d'écran initiale montre l'interface de saisie des paramètres, confirmant la simplicité et la clarté du formulaire proposé par le frontend.

Une fois la requête envoyée, l'application génère automatiquement un itinéraire structuré, comme illustré par les captures suivantes. Les résultats affichés indiquent clairement la ville sélectionnée ainsi que la durée du séjour. Chaque journée est ensuite organisée en trois créneaux distincts, correspondant au matin, à l'après-midi et au soir. Cette structuration temporelle valide le bon fonctionnement de l'algorithme de planification implémenté côté backend.

Navigation

Go to:

- Itinerary Generator
- AI Chatbot

Morocco Smart Guide – Tourism AI
MA

Generate a personalized itinerary

Main city

Duration (days)

- +

Budget

Interests

Constraints

Language

Generate itinerary

Pour la première journée, les activités proposées incluent notamment la visite du Palais Bahia le matin, le Jardin Majorelle l’après-midi et la place Jemaa el-Fna en soirée. Les horaires sont explicitement indiqués, et chaque activité est accompagnée d’une description détaillée ainsi que de conseils pratiques. Ces éléments démontrent l’intégration correcte du mécanisme de Retrieval-Augmented Generation, les descriptions étant enrichies par le modèle de langage à partir des données issues de la base vectorielle.

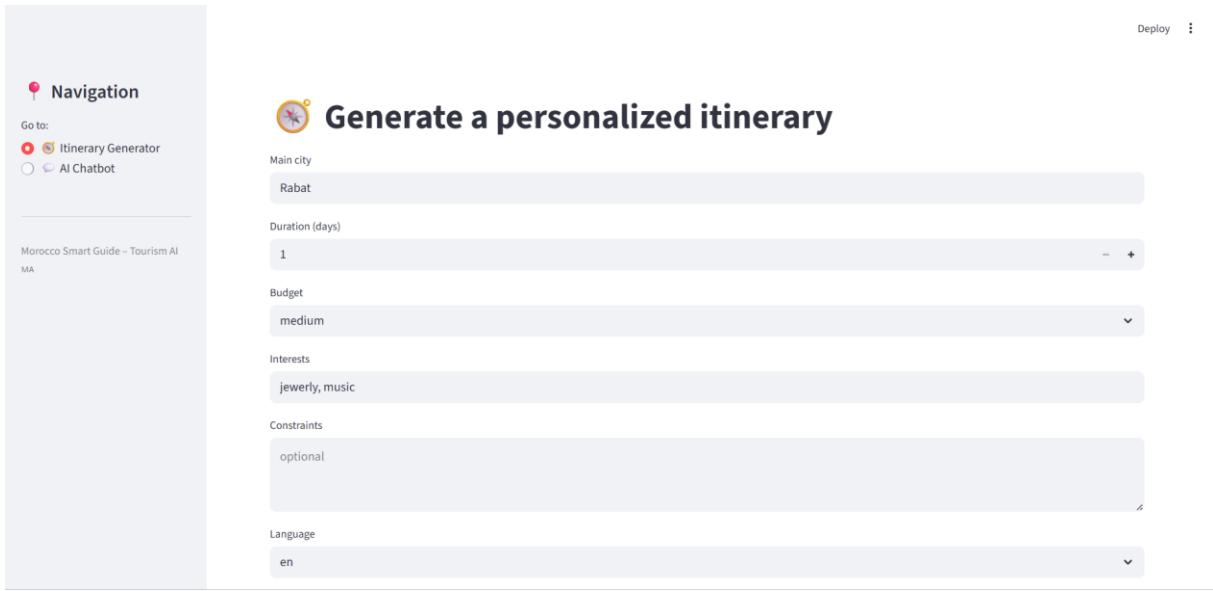
Day 2 ↴

Morning	Afternoon	Evening
🕒 09:00 – 12:00 Saadian Tombs Discover the hidden beauty of the Saadian Tombs, an intricately decorated burial site filled with mausoleums and colorful tile work. Recommended duration: 2 hours. Tips: Wear modest clothing and remove your shoes before entering.	🕒 14:00 – 17:00 Souk Sebbaghine Indulge in the flavors of Marrakech at the Souk Sebbaghine, a traditional market filled with spices, olives, and other local delicacies. Recommended duration: 3 hours. Tips: Haggle for the best prices and try new foods.	🕒 18:00 – 20:00 Jemaa el-Fna Food Stalls Savor the authentic flavors of Moroccan cuisine at the food stalls around Jemaa el-Fna square. Recommended duration: 2 hours. Tips: Try the famous Moroccan tagine and mint tea.

La deuxième journée présente également une organisation cohérente, avec la visite des Tombeaux Saadiens le matin, du souk Sebbaghine l’après-midi et des stands de restauration de Jemaa el-Fna en soirée. La continuité logique entre les activités, la prise en compte des durées recommandées et le respect des créneaux horaires confirment la fiabilité de la méthodologie adoptée pour la génération des itinéraires.

Dans l’ensemble, ce premier test valide la capacité de l’application à produire un itinéraire personnalisé, structuré et directement exploitable à partir de paramètres simples. Les captures d’écran associées illustrent non seulement la qualité de l’interface utilisateur, mais également la cohérence des résultats générés, ce qui constitue un indicateur positif du bon fonctionnement global du système.

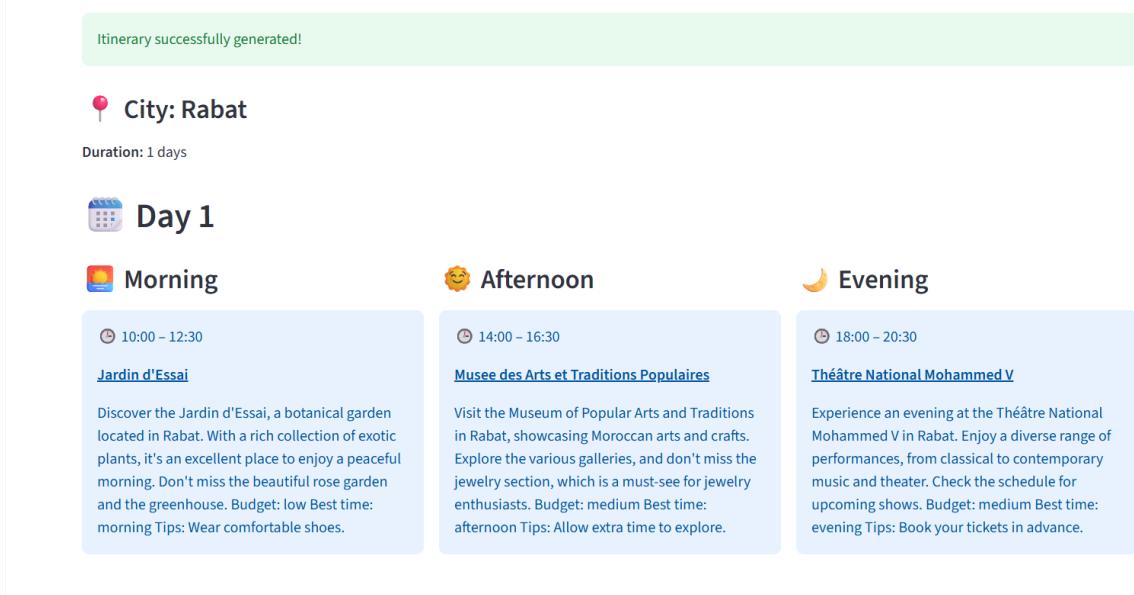
Deuxième test sur Rabat



The screenshot shows a user interface for generating a personalized itinerary. On the left, there's a sidebar with navigation links like 'Itinerary Generator' and 'AI Chatbot'. The main area has fields for 'Main city' (Rabat), 'Duration (days)' (1), 'Budget' (medium), 'Interests' (jewelry, music), 'Constraints' (optional), and 'Language' (en). A 'Deploy' button is at the top right.

Le second test porte sur la génération d'un itinéraire touristique pour la ville de Rabat. Ce scénario vise à évaluer la capacité du système à s'adapter à une autre ville, à une durée de séjour plus courte et à des centres d'intérêt différents. Dans ce cas, les paramètres fournis correspondent à un séjour d'une seule journée, avec un budget moyen et des centres d'intérêt orientés vers la joaillerie et la musique. La langue de génération a également été définie en anglais, afin de maintenir une cohérence avec le premier test et de simuler un usage par un visiteur international.

La capture d'écran associée à la saisie des paramètres met en évidence la flexibilité de l'interface, qui permet de modifier facilement la ville, la durée du séjour et les centres d'intérêt. Une fois la requête validée, l'application confirme la génération réussie de l'itinéraire, ce qui démontre le bon fonctionnement du pipeline de traitement côté backend, indépendamment de la ville sélectionnée.



The itinerary summary says 'Itinerary successfully generated!'.

City: Rabat
Duration: 1 days

Day 1

Morning	Afternoon	Evening
Jardin d'Essai Discover the Jardin d'Essai, a botanical garden located in Rabat. With a rich collection of exotic plants, it's an excellent place to enjoy a peaceful morning. Don't miss the beautiful rose garden and the greenhouse. Budget: low Best time: morning Tips: Wear comfortable shoes.	Musee des Arts et Traditions Populaires Visit the Museum of Popular Arts and Traditions in Rabat, showcasing Moroccan arts and crafts. Explore the various galleries, and don't miss the jewelry section, which is a must-see for jewelry enthusiasts. Budget: medium Best time: afternoon Tips: Allow extra time to explore.	Théâtre National Mohammed V Experience an evening at the Théâtre National Mohammed V in Rabat. Enjoy a diverse range of performances, from classical to contemporary music and theater. Check the schedule for upcoming shows. Budget: medium Best time: evening Tips: Book your tickets in advance.



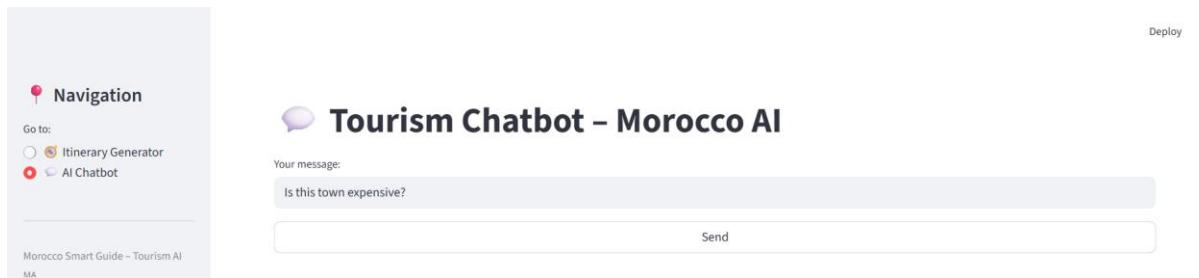
Les résultats générés affichent clairement la ville de Rabat ainsi que la durée du séjour. L'itinéraire proposé est organisé sur une seule journée, répartie de manière cohérente en trois créneaux temporels. Le matin est consacré à la visite du Jardin d'Essai, activité adaptée à un début de journée calme et correspondant aux préférences culturelles et esthétiques. L'après-midi inclut la visite du Musée des Arts et Traditions Populaires, mettant en valeur le patrimoine culturel et artisanal de la ville, avec une attention particulière portée aux centres d'intérêt liés à la joaillerie. Enfin, la soirée propose une activité au Théâtre National Mohammed V, en adéquation avec l'intérêt exprimé pour la musique et les spectacles.

L'ensemble des activités est accompagné d'horaires précis, de descriptions détaillées et de conseils pratiques, ce qui confirme la bonne intégration du mécanisme de recherche sémantique et de génération contrôlée. La cohérence temporelle et thématique de l'itinéraire généré démontre la capacité du système à adapter ses recommandations à des contextes variés, tant en termes de durée que de centres d'intérêt.

Ainsi, ce second test valide la robustesse et la générnicité de la solution proposée. Les captures d'écran associées illustrent la capacité de l'application à produire des itinéraires personnalisés pour différentes villes marocaines, tout en respectant les contraintes définies et en maintenant une structure claire et exploitable.

Dernier test sur le chatbot

Le dernier test porte sur l'évaluation du chatbot touristique intégré à l'application. Ce test vise à vérifier la capacité du système à fournir des réponses pertinentes, contextualisées et cohérentes à partir d'une interaction conversationnelle continue. Contrairement à la génération d'itinéraires, ce mode d'utilisation repose sur des échanges successifs permettant d'explorer une ville ou un thème de manière progressive.



La première interaction consiste en une requête générale portant sur la ville de Tanger. La capture d'écran associée montre que le chatbot est capable de produire une réponse détaillée présentant la ville, tout en proposant des recommandations culturelles variées. Les informations fournies concernent notamment des musées, des circuits thématiques et des lieux emblématiques, ce qui démontre la capacité du système à exploiter efficacement la base de connaissances indexée et à restituer des contenus riches et structurés.



💬 History

💡 You: Talk to me about Tangier

🤖 Assistant: Tangier is a vibrant and historic city located in Morocco. If you're interested in communication history, I'd recommend visiting the Musée des Communications. Here, you can explore the evolution of communication methods in Tangier, from telegraphs and telephones to radios and the internet. The museum also highlights Tangier's role as a communication hub.

For those who enjoy shopping and exploring scents, the Circuit des Parfums is a must-visit. You'll get to discover the unique fragrances of Tangier through its spice markets and flower markets. Don't forget to create your own souvenir blend!

History buffs may also appreciate the Musée de la Chasse, which showcases historical hunting weapons and trophies, as well as the evolution of hunting practices in the Tangier region during the 19th and 20th centuries.

If you're interested in international relations and diplomacy, the Musée de la Diplomatie is a free attraction that offers an insight into Tangier's past as an international city and the home of numerous foreign legations during the 19th and 20th centuries.

Lastly, literature enthusiasts can follow the Circuit Littéraire to visit the places frequented by famous writers who lived in Tangier, such as Paul Bowles, William S. Burroughs, and Allen Ginsberg. A specialized guide and some pre-reading of these authors' works are recommended to enhance your experience.

💡 You: Is this town expensive?

🤖 Assistant: Based on the context provided, Tangier offers a variety of shopping experiences with different budgets. The Medina de Tanger and Petit Socco are great options for those on a low budget, where you can find local artisanal products, spices, and explore the historic sites.

However, for those looking for international brands and modern shopping experiences, the Rue de la Liberté in Tangier or the Carré Eden Mall in Marrakech may be more expensive options. As always, it's essential to negotiate prices and haggle when shopping in local markets to get the best deals.

Par la suite, une question de suivi est posée concernant le coût de la vie dans la ville, sans reformuler explicitement le contexte. La réponse générée montre que le chatbot prend en compte l'historique de la conversation, en fournissant une analyse des niveaux de budget adaptés à différents types de visiteurs. Les recommandations distinguent clairement les zones accessibles à faible budget et celles associées à des dépenses plus élevées, tout en intégrant des conseils pratiques tels que la négociation dans les marchés locaux.

Ce comportement met en évidence le bon fonctionnement du mécanisme de Retrieval-Augmented Generation combiné à une mémoire conversationnelle. Les réponses produites s'appuient sur des informations issues de la base vectorielle tout en maintenant une continuité logique entre les échanges. Cette approche permet de se distinguer d'un simple agent conversationnel générique, en garantissant que les réponses restent ancrées dans des données spécifiques au contexte touristique marocain.

Ainsi, ce test valide la capacité du chatbot à fournir une assistance touristique fiable et contextualisée, aussi bien pour des questions générales que pour des demandes plus ciblées. Les captures d'écran associées illustrent la fluidité de l'interaction et la pertinence des réponses générées, confirmant l'efficacité de l'architecture RAG mise en œuvre dans le cadre du projet.

VIII. Difficultés rencontrées et solutions apportées

Le développement de l'application Morocco Smart Guide a soulevé plusieurs difficultés techniques et conceptuelles, liées à la gestion des données, aux performances du système et à l'intégration des mécanismes d'intelligence artificielle. Ces difficultés ont permis d'identifier certaines limites de la solution proposée et ont conduit à la mise en place de choix techniques adaptés.

Une première difficulté concerne l'indexation des données touristiques au sein de la base vectorielle FAISS. Bien que chaque fichier CSV associé à une ville contienne environ

trois cents lieux touristiques, l'analyse de l'index FAISS a révélé un nombre de documents significativement inférieur à celui attendu. En effet, un script de vérification a montré que seules certaines entrées étaient effectivement présentes dans l'index, avec une forte disparité entre les villes. Cette situation peut s'expliquer par des problèmes liés à la qualité ou à la structure des données d'entrée, notamment des lignes incomplètes, des identifiants manquants ou des champs essentiels non renseignés, entraînant l'exclusion de certains lieux lors du processus de chargement et de transformation en documents. Cette difficulté met en évidence l'importance d'un contrôle plus strict des données en amont et d'une validation approfondie du pipeline d'ingestion.

Une autre difficulté majeure concerne les performances du système, en particulier lors de la génération des itinéraires. L'appel au modèle de langage via l'API Hugging Face, combiné à la récupération des documents et à la planification algorithmique, entraîne un temps de réponse relativement élevé. Cette latence est principalement due à la taille du modèle de génération utilisé ainsi qu'au caractère synchrone des appels effectués. Bien que ce choix garantisse une meilleure qualité des réponses produites, il impacte l'expérience utilisateur et limite la fluidité de l'interaction avec l'application.

Par ailleurs, l'ergonomie globale de l'application constitue un point perfectible. L'interface Streamlit, bien que fonctionnelle et simple à mettre en œuvre, présente certaines limites en matière de réactivité et de personnalisation avancée de l'expérience utilisateur. L'absence d'indicateurs précis sur l'état d'avancement de la génération ou de mécanismes de chargement plus interactifs peut rendre l'attente moins intuitive pour l'utilisateur, en particulier lors des requêtes complexes.

D'autres difficultés ont également été rencontrées lors de l'intégration des modèles de langage. La variabilité des formats de réponse générés par l'API Hugging Face a nécessité la mise en place de mécanismes robustes d'extraction et de validation du contenu, notamment pour garantir la conformité des réponses JSON utilisées dans la génération d'itinéraires. De plus, la gestion de la compatibilité entre différentes versions de LangChain a requis des adaptations spécifiques afin d'assurer la stabilité du système.

Enfin, la gestion de la mémoire conversationnelle du chatbot a posé des défis liés à la persistance des sessions. Le choix d'une mémoire stockée uniquement en mémoire vive simplifie l'implémentation, mais limite la continuité des échanges à long terme et ne permet pas une reprise de session après redémarrage de l'application. Cette contrainte souligne la nécessité d'envisager des solutions de persistance plus robustes dans une version future.

Dans l'ensemble, ces difficultés ont permis de mieux comprendre les limites techniques et conceptuelles d'un système d'intelligence artificielle intégré. Elles ouvrent également la voie à plusieurs pistes d'amélioration, notamment en matière de nettoyage des données, d'optimisation des performances, d'amélioration de l'ergonomie et de renforcement de la robustesse globale du système.

Conclusion

Ce projet a permis de concevoir et de mettre en œuvre une application d'intelligence artificielle dédiée à l'assistance touristique, intégrant des mécanismes avancés de recherche sémantique, de génération contrôlée de contenu et de planification algorithmique. À travers le développement de Morocco Smart Guide, une solution complète a été proposée afin de répondre aux problématiques liées à la personnalisation et à la structuration de l'information touristique dans un contexte réel et pertinent.

L'architecture adoptée, reposant sur une séparation claire entre le frontend, le backend et la couche d'intelligence artificielle, a permis d'assurer la modularité et la cohérence du système. L'utilisation d'une base vectorielle FAISS combinée à un mécanisme de Retrieval-Augmented Generation a contribué à améliorer la fiabilité des recommandations produites, tout en limitant les limites inhérentes aux modèles de langage généralistes utilisés de manière isolée. Par ailleurs, l'intégration d'un algorithme de planification explicite a garanti la production d'itinéraires structurés, cohérents et directement exploitables.

Les tests réalisés ont permis de valider le bon fonctionnement de l'application dans différents scénarios, couvrant plusieurs villes marocaines et des profils d'utilisateurs variés. Les résultats obtenus démontrent la capacité du système à s'adapter aux contraintes fournies et à proposer des recommandations pertinentes, aussi bien sous forme d'itinéraires personnalisés que via une assistance conversationnelle contextuelle.

Néanmoins, certaines limites ont été identifiées, notamment en ce qui concerne la complétude de l'indexation des données, les performances lors de la génération et l'ergonomie de l'interface utilisateur. Ces limites mettent en évidence des axes d'amélioration possibles, tels que l'optimisation du pipeline d'ingestion des données, l'introduction de mécanismes asynchrones pour réduire la latence, ou encore l'amélioration de l'expérience utilisateur par une interface plus réactive et plus informative.

En perspective, ce projet constitue une base solide pour des évolutions futures, incluant l'enrichissement de la base de connaissances, l'intégration de sources de données externes, la persistance des sessions de conversation et le déploiement du système à plus grande échelle. Ainsi, Morocco Smart Guide illustre de manière concrète l'apport des systèmes d'intelligence artificielle intégrée dans le domaine du tourisme intelligent, tout en mettant en évidence les défis et opportunités associés à leur mise en œuvre.