PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
Ministry of Higher Education and Scientific Research
Mohamed Khider University – BISKRA
Faculty of Exact Sciences, Natural and Life Sciences

Computer Science Department

N° d'ordre : Startup_05/GLSD/M2/2024

THESIS

Submitted in fulfillment of the requirements for the Master's degree in

Computer Science

Option: Software engineering and distributed systems (GLSD)

A Travel Recommendation System Based On Machine Learning Techniques

---

By:

Salhaoui Souria

supported the 25 june 2024 , Members of the jury:

- Ben seghier Nadia	MCA	President
- Abdelli Belkacem	MCB	Rapporteur
- Touil Keltoum	MAA	Examiner

Academic year : **2023 – 2024**


بسم الله الرحمن الرحيم

Abstract

The travel recommendation system leverages technologies like machine learning, artificial intelligence, and big data analysis to understand user behavior and predict preferences, enabling personalized and effective travel suggestions. This system enhances traveler experiences by providing tailored recommendations, saving time in finding suitable destinations and activities, and raising awareness of new tourist spots. Developing such a system poses technical and strategic challenges but offers significant opportunities to improve traveler satisfaction.

A dedicated travel web application has been developed based on these recommendation and machine learning techniques. It allows users to discover major tourist attractions, access information about hotels and restaurants, and book various trips by connecting with travel agencies. The application also supports personal advertisements, expanding user choices when searching. Its key feature is recommending numerous tourist destinations based on user preferences, including details like city, price, cost, and time. Additionally, it offers quick search options for various user-required facilities.

key words : Machine Learning , Recommendation System , Travel , Tourist , Web Application

الملخص

يستفيد نظام التوصية بالسفر من تقنيات مثل التعلم الآلي والذكاء الاصطناعي وتحليل البيانات الضخمة لفهم سلوك المستخدم والتنبؤ بالتفضيلات، مما يتيح اقتراحات سفر مخصصة وفعالة. يعزز هذا النظام تجارب المسافر من خلال تقديم توصيات مخصصة، وتوفير الوقت في العثور على وجهات وأنشطة مناسبة، وزيادة الوعي بالمواقع السياحية الجديدة.

يفرض تطوير مثل هذا النظام تحديات تقنية واستراتيجية ولكنه يوفر فرصًا كبيرة لتحسين رضا المسافر.

تم تطوير تطبيق ويب مخصص للسفر بناءً على تقنيات التوصية والتعلم الآلي هذه. يسمح للمستخدمين باكتشاف مناطق الجذب السياحي الرئيسية، والوصول إلى معلومات حول الفنادق والمطاعم، وحجز رحلات مختلفة من خلال الاتصال بوكالات السفر. يدعم التطبيق أيضًا الإعلانات الشخصية، مما يوسع خيارات المستخدم عند البحث. تتمثل ميزته الرئيسية في التوصية بالعديد من الوجهات السياحية بناءً على تفضيلات المستخدم، بما في ذلك التفاصيل مثل المدينة والسعر والتكلفة والوقت. بالإضافة إلى ذلك، فإنه يوفر خيارات بحث سريعة لمختلف المرافق المطلوبة من قبل المستخدم.

االكلمات المفتاحية : التعلم الآلي، نظام التوصية، السفر، السياحة، تطبيق الويب

Acknowledgements

In the name of Allah, the Most Gracious, the Most Merciful. Praise be to Allah who enabled me to complete this work.

I extend my sincere thanks, appreciation and love to all my esteemed professors, and I would like to thank**Dr. Abdelli Belkacem** , may Allah protect him and prolong his life, for his generous supervision of this thesis and his guidance to complete it.

I express my gratitude to all my esteemed professors throughout my academic career.

And to all my colleagues and friends who stood by me, I extend my thanks.

Finally, I would like to extend my deepest gratitude to myself for my perseverance, determination and hard work to provide the best.

Dedication

Whoever said, “I got her,” the journey was not short, nor should it be. The dream was not close, nor was the path lined with facilities. After an academic journey that lasted for years, here I am today, standing on the threshold of my graduation, reaping the fruits of my labor and raising my hat with pride. Praise be to God for love, thanks and gratitude, thanks to which I am here today looking at a long-awaited dream that has become a reality that I am proud of. .... I dedicate this success

To the pillar of the family, to the good souls, to those who taught me that the world is a struggle and its weapon is knowledge and knowledge, to those who instilled in our souls good morals and kindness, my grandfather “Bekhtaoui Kamal” and my grandmother “Habib Khaira” from your first granddaughter.

To my dear, dear grandmother, “Dalouli Kulthum”

To the one who adorned my name with the most beautiful titles, who supported me without limits and gave me for free, to my support, my strength, and my refuge after God, who was my first supporter and the source of my success, to my pride and pride, my dear father “Othman.”

To the one who made Paradise under her feet, and whose heart embraced me before her hand, and made adversity easy for me with her supplications, to the candle that was for me on dark nights as my path lamp, to the one who spread roses for me on my path with her tenderness, my dear mother, “Bakhtaoui Fatima Al-Zahra.”

To those about whom it was said, “We will support you with your brother.” To my princesses, “Malak,” Kulthum, “Farah,” Soudjod, “May you always give me happiness that fills my heart.”

To the man of the house after my father, “my brother.” May God keep you as a steadfast side for me

To those who were called the second mother, my aunts, “Halima,” “Khadija,” “Fatiha”

To the one who was an aunt and companion for ten years, “Hind”

To those who were a piece of the mother, to my dear uncle “Amin”, to the expatriate who has been away for a long time and whose place remains calling by his name, my dear uncle “Abdelhak”

To ten years of age, soul friends and companions “Feriel” “Karima” “Maissa”

To everyone who has been a help and support on this path... I present to you the fruit of this achievement and the fruit of my success, which I have always wished for. Here we are today, completing the first of its fruits, hoping for God Almighty to benefit me with what He has taught me, to teach me what I do not know, and to make it an excuse for me, not against me.

الإهداء

من قال أنا لها " نالها " لم تكن الرحلة قصيرة ولا ينبغي لها أن تكون . لم يكن الحلم قريبا و لا الطريق كان محفوفا بالتسهيلات و بعد مسيرة دراسية دامت سنوات ها أنا اليوم أقف على عتبة تخرجي أقطف ثمار تعبي و أرفع قبعتي بكل فخر الحمدلله حبا و شكرا و امتنانا الذي بفضله ها أنا اليوم أنظر الى حلم طال انتظاره و قد أصبح واقعا أفتخر به أهدي هذا النجاح ....

الى عمود العائة الى الأرواح الطيبة الى من علموني أن الدنيا كفاح و سلاحها العلم و المعرفة الى من غرسو في أرواحنا مكارم الأخلاق و الطيبة جدي "بختاوي كمال" و جدتي "حبيب خيرة" من حفيدتكم الأولى

الى عزيزتي غاليت قلبي جدتي " دلولي كلثوم"

الى الذي زين إسمي بأجمل الألقاب من دعمني بلا حدود و أعطاني بلا مقابل الى سندي و قوتي و ملاذي بعد الله يا من كان داعمي الأول و مصدر نجاحي الى فخري و اعتزازي أبي الغالي "عثمان"ا

الى من جعل الله الجنة تحت أقدامها و احتضنني قلبها قبل يدها و سهلت لي الشداند بدعائها الى الشمعت التي كانت لي في اليالي المظلمة مصباح دربي الى من أفرشت لي طريقي ورودا بحنانها أمي الغالية "بختاوي فاطمة الزهراء"

الى من قيل فيهم سنشد عضدك بأخيك الى أميراتي قلبي "ملاك" كلثوم" "فرح" سجود" دمتم لي سعادتا تملأ قلبي

الى رجل البيت بعد أبي "أخي " أدامك الله ضلعا ثابتا لي

الى من قيل فيهم الأم الثانية خالاتي "حليمة" "خديجة" " فتيحة"

الى من كانت خالة و رفيقة درب و عشرة عمر "هند"

الى من كانو قطعة من الأم الى الغالي خالي "أمين" الى المغترب من طال غيابه و ظل مكانه ينادي بإسمه خالي الغالي "عبد الحق"

الى عشرة العمر صديقات الروح رفيقات الدرب "فريال" " كريمة" "رميسة"

الى من كانو رائحة من الأب عماتي و أعمامي

لكل من كان عونا و سندا في هذا الطريق ... أهديكم ثمرة هذا الإنجاز و ثمرة نجاحي الذي لطالما تمنيته ها أنا اليوم أتممت أول ثمراته راجية من الله تعالى أن ينفعني بما علمني و أن يعلمني ما أجهل و يجعله حجة لي لا علي.

Table of Contents

List of Figures

List of Tables

General Introduction

0.1 context

In the contemporary digital age, the travel industry is undergoing a profound transformation fueled by advancements in technology. One of the most significant developments in this domain is the emergence of travel recommendation systems powered by machine learning. These systems are designed to assist travelers in making informed decisions about their trips by providing personalized suggestions tailored to their preferences and behaviors. [7]

A travel recommendation system leverages machine learning algorithms to analyze vast amounts of data from various sources such as user reviews, social media, booking history, and demographic information. By processing this data, the system can identify patterns and trends, enabling it to predict and recommend destinations, accommodations, activities, and itineraries that align with the user's interests and requirements.

Machine learning, a subset of artificial intelligence, plays a pivotal role in enhancing the accuracy and relevance of travel recommendations. Key techniques used in these systems include collaborative filtering, which makes recommendations by analyzing the preferences of similar users, and content-based filtering, which suggests travel options based on the specific attributes of items a user has shown interest in previously. Additionally, natural language processing (NLP) is employed to analyze user-generated content such as reviews and social media posts, while deep learning models capture complex patterns and relationships in diverse data types. Context-aware recommendations further enhance relevance by incorporating information such as the user's current location, time of travel, weather conditions, and travel companions.

The benefits of machine learning-based travel recommendation systems are manifold. These algorithms offer highly personalized travel suggestions, enhancing user satisfaction by catering to individual preferences and needs. They save time for travelers by filtering out irrelevant options, simplifying the decision-making process, and continuously improving through learning from user interactions. Moreover, by analyzing trends and patterns in travel data, these systems provide valuable insights to travel agencies and service providers, enabling them to tailor their offerings more effectively.

Machine learning-based travel recommendation systems are revolutionizing the way people plan and experience their travels. By harnessing the power of advanced algorithms and vast datasets, these systems provide personalized, efficient, and insightful recommendations, paving the way for more enjoyable and memorable travel experiences.

0.2 Problematic

Tourists face many problems related to the ideal tourist place that matvhes his oreferences destinations and prices. One of the most prominent of these problems is not knowing the ideal tourist places, as most of the offers that are proposed are according to the desires of travel agencies, i.e. they give them a limited scope of choice. The high costs of travel make it difficult for many people to travel to their favorite destinations. Also, many tourists want to go to new places and explore new areas, but they do not have enough information

0.3 Objectives and solution

The goal of my project, a machine learning-based travel recommendation system, is to create an efficient and user-friendly web application and platform that helps travelers make informed decisions about their travel. By leveraging advanced machine learning algorithms, the system aims to analyze massive amounts of data from various sources, such as user reviews, to identify patterns and trends. This analysis will enable the system to predict and recommend destinations, accommodations, activities, and travel itineraries that match the preferences and requirements of the individual user. The ultimate goal is to improve the travel planning experience by providing personalized recommendations that save time, increase satisfaction, and ensure a memorable and enjoyable trip for each user. This can be explained in the following points:

- Knowing customers' desires and budgets through evaluation
- Knowing the best tourist destinations
- Providing tourism offers that are most closely related to customers' desires
- Trying to know prices and improve them according to each customer's budget

1
Recommendation System

1.1 Introduction

The evolution of recommender systems technology has been greatly influenced by the increasing significance of the Web as a platform for electronic and business transactions. A key driver in this evolution is the seamless feedback mechanism facilitated by the Web, allowing users to express their preferences easily. Take, for instance, the scenario of a content provider like Netflix, where users can effortlessly provide feedback through simple clicks. Typically, feedback is given through ratings, where users assign numerical values from a specific evaluation system (e.g., a five-star rating system) to indicate their preferences. Other forms of feedback, though less explicit, are equally valuable in the Web-centric environment. For example, a user's purchase or browsing activity can serve as an implicit endorsement for an item. Online retailers like Amazon.com commonly leverage such feedback, making data collection effortless for customers.

Recommender systems aim to utilize these diverse data sources to infer customer interests. The recipient of the recommendation, known as the user, and the recommended product, referred to as an item, form the basis of recommendation analysis. This analysis often relies on past interactions between users and items, as previous preferences are indicative of future choices. Knowledge-based recommender systems, however, offer recommendations based on user-specified requirements rather than historical data.

The fundamental principle underlying recommendation algorithms is the existence of significant relationships between user and item interactions. For instance, a user interested in historical documentaries is more likely to prefer similar content over unrelated genres like action movies. By identifying correlations between different item categories or individual items, recommender systems can make more accurate suggestions. These dependencies are learned from the ratings matrix in a data-driven approach, enabling predictions for target users.

The effectiveness of recommendation algorithms is enhanced by the availability of a larger number of rated items for a user, enabling more reliable predictions about their future behavior.

Various learning models, such as collaborative filtering, leverage collective user behavior to create cohorts of similar users interested in comparable products. These cohorts' preferences guide recommendations for individual members, illustrating the concept of neighborhood models within the broader class of collaborative filtering models.

While the description above focuses on a basic family of recommendation algorithms, known as neighborhood models, the realm of recommender systems encompasses diverse and data-rich approaches. Content-based recommender systems, for instance, prioritize item attributes and user ratings to predict preferences. By modeling user interests based on item properties, these systems offer personalized recommendations tailored to individual preferences [8].

The rapid expansion of digital information and internet users has led to a significant challenge of information overload, impeding timely access to relevant content online. While search engines like Google, DevilFinder, and Altavista have partially addressed this issue, the lack of prioritization and personalization in information retrieval systems has fueled a growing demand for recommender systems. These systems act as information filters, tackling information overload by extracting crucial data fragments from vast dynamically generated content based on user preferences, interests, or observed behaviors regarding items. By leveraging user profiles, recommender systems can predict user preferences accurately.

Recommender systems offer benefits to both service providers and users by reducing transaction costs in online shopping environments and enhancing decision-making processes and quality. Particularly in e-commerce settings, these systems boost revenues by effectively promoting more products to users. In scientific libraries, recommender systems empower users to explore beyond traditional catalog searches. Therefore, the importance of employing efficient and precise recommendation techniques within a system that delivers relevant and reliable recommendations for users cannot be overstated. [9]. Recommender systems aim to predict preferences or ratings using content-based filtering, collaborative filtering, or knowledge-based recommender system. Content-based recommender systems work on the concept of item features and user likes based on past history. Collaborative recommendation systems work on the concept of modifying information based on ratings from similar users. A knowledge-based recommendation system works on the concept of users' preferences and needs to recommend results. A hybrid method is a combination of two or more methods together depending on the complexity of the problem. This paper looks more closely at travel recommendation systems, why they are needed, the pros and cons of current travel recommendation systems, and a new approach to overcoming the pros and cons of the current travel recommendation system. Travel and tourism are activities closely linked to the traveler's personal preferences and interests. Today, many trip advisory web applications infuse recommender systems. This recommendation system creates or imitates the interaction between human travel agents [10].

1.2 Definition

A recommender system, as defined by Bobadilla et al. (2013), is a collection of programs designed to suggest the most relevant items to specific users by predicting their interest in an item based on information about the items, users, and interactions between them. On the other hand, a recommendation system is an AI-driven technology that leverages machine learning and data analysis to offer personalized recommendations to users based on their behavior, preferences, and past interactions. These systems find extensive application across industries like e-commerce, media, entertainment, travel, and gaming to boost user engagement, drive sales, and enhance customer satisfaction. Users of these systems can range from individuals to businesses seeking various items like books, job opportunities, or business partners. The items recommended can be products or services such as books, movies, or mobile service packages. as shown in Fig. 1.1. [1].

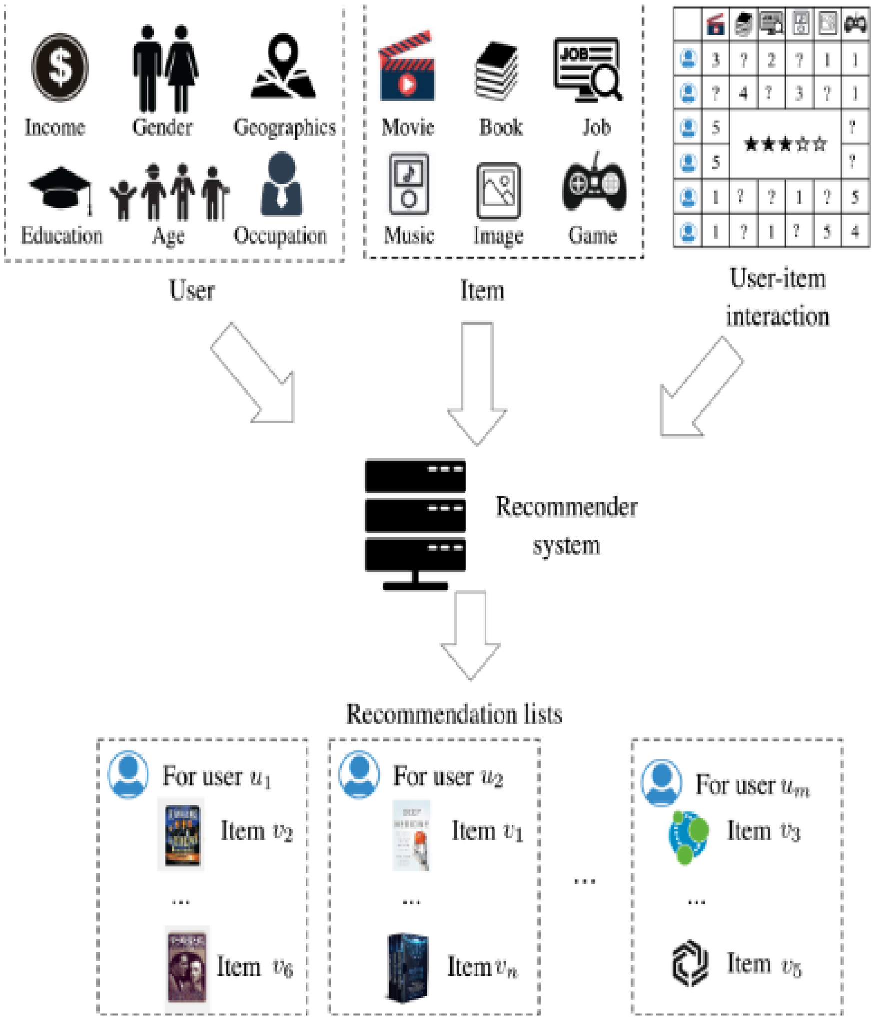


Figure 1.1: A graphical illustration of a recommender system [1]

1.3 Recommendation Techniques

Data mining is a technique for deriving useful information by discovering correlations and patterns between data based on data analysis in large datasets . It analyzes the information of the item, makes it possible to recommend items similar to the item to the user, and creates a similar user group among users to identify the client/visitor click stream data matching the user group. It can also recommend customized browsing options to meet the needs of specific users . Various data mining analysis techniques are used for such recommendations. as shown in Figure 1.2 is a visual summary of the techniques mainly used in the recommendation system to be described in this section. Furthermore,and Figure 1.3 shows a typical data mining process. [2]

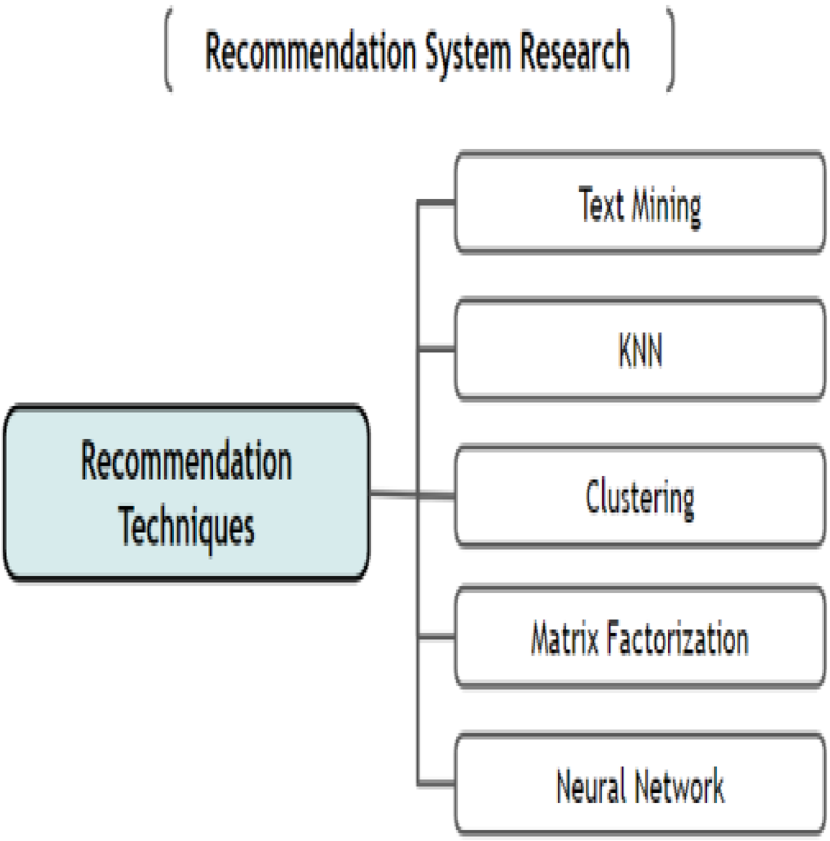


Figure 1.2: Technology mainly used in recommendation system. system [2]

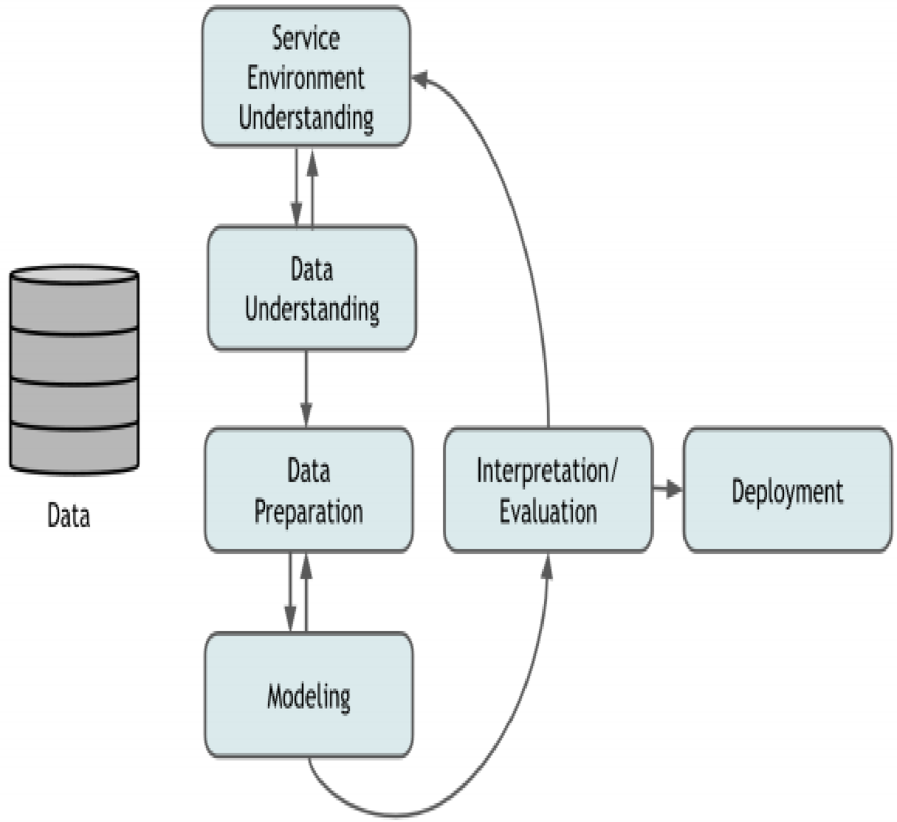


Figure 1.3: Typical data mining process. system [2]

1.3.1 Text Mining

Text mining is a technique for discovering useful text information by extracting text related information from data. With the recent development of natural language processing technology, semantically important information has been extracted from the corresponding text , When natural language processing is used in some text analysis processes, there is a tendency to analyze texts based on the frequency of words, so there is a limit to understanding semantics , For this reason, in order to accurately grasp the meaning of the text, the ontology, which defines the common vocabulary of items and organizes the meaning by constructing the conceptual schema of the text domain, started to be used. This text mining is used to recommend similar items by performing semantic analysis of item information in the Content-Based Filtering recommendation model . In addition, the Collaborative Filtering recommendation model evaluates the semantic knowledge of information data between users, enabling item recommendation with similarity . Figure 1.4 is a visual summary of a typical text mining process [2]

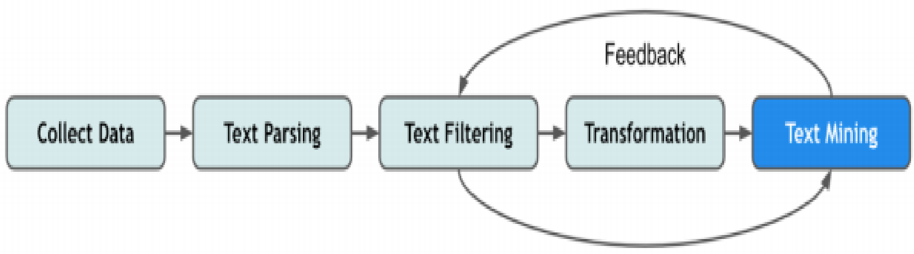


Figure 1.4: Typical text mining process. [2]

Text mining technology plays a crucial role in enhancing communication between humans and computers by advancing towards context awareness. Fuzzy linguistic modeling (FLM) is a text mining technique that incorporates fuzzy logic into natural language processing to analyze language meaning through fuzzy subsets. When integrated into recommendation systems, FLM enables the identification of multilingual contexts for items, especially useful when user preferences are unclear or data is insufficient. By analyzing text data between items, FLM can supplement inadequate preference data by establishing preference relationships.

Within Content-Based Filtering recommendation models, the primary text mining technique employed is Term Frequency – Inverse Document Frequency (TF-IDF), which assigns weight based on the frequency of specific text occurrences. This method represents document text components as vectors and determines term importance by calculating the relative frequency of a word in a document using the TF-IDF weight function. Text mining techniques find widespread application in sectors such as healthcare, education, tourism, and academic services, demonstrating their versatility and impact across various domains. [2]

1.3.2 KNN (K-Nearest Neighbor)

The K-Nearest Neighbor (KNN) algorithm is utilized to classify datasets by identifying the K-nearest neighbors of a test tuple and train tuple. This classification is based on proximity, comparing the similarity between data items through distance-based weighting. Common similarity measures like Euclidean distance, cosine similarity, and Pearson correlation are employed to assess similarities. When integrated into a recommendation system, KNN can analyze user search patterns to predict future preferences. By examining user behavior data such as web server logs and clickstream data, KNN can categorize items aligning with user tastes and recommend relevant items accordingly.

However, a study by Jannach et al. highlighted limitations in the performance of recommendation models utilizing the KNN algorithm. Challenges arise from the need to select an optimal value for K, impacting model performance and leading to biases. Moreover, KNN's efficiency diminishes when analyzing large input data sets. To address this, dimensionality reduction techniques are employed to transform data into a more manageable form without losing critical information. Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) are commonly utilized for dimensionality reduction, aiming to enhance the efficiency of data analysis processes, particularly in scenarios with extensive input data. [2]

1.3.3 Clustering

Clustering is an algorithm utilized to categorize data into distinct clusters, offering a method with low redundancy and ambiguity that is highly favored in recommendation systems. Various clustering techniques are employed in recommendation systems, with K-means clustering being a prominent choice. K-means clustering involves grouping data around the mean after defining the number of clusters (K), assigning data points to the nearest cluster based on similarity calculations, and iteratively updating cluster centers.

However, K-means clustering faces scalability challenges when the number of users and items increases, leading to decreased calculation speed. To address this, Gong implemented user and item clustering using an inter-user clustering technique based on user ratings, enhancing personalized recommendations. By identifying cluster groups similar to the target user through item similarities, scalability and sparsity issues inherent in traditional Collaborative Filtering approaches were mitigated.

Clustering plays a significant role in Collaborative Filtering recommendation models and is extensively studied in recommendation systems across sectors like tourism, education, and e-commerce. Its ability to group similar data points efficiently contributes to enhancing recommendation accuracy and personalization in diverse application domains. [2]

1.3.4 Matrix Factorization

Clustering is a fundamental algorithm used in recommendation systems to categorize data into distinct clusters, offering a low-redundancy and unambiguous approach. Among various clustering techniques, K-means clustering stands out as a popular choice. This algorithm involves grouping data around the mean after specifying the number of clusters (K), assigning data points to the nearest cluster based on similarity calculations, and iteratively updating cluster centers.

However, scalability issues can arise with K-means clustering as the number of users and items grows, leading to reduced calculation speed. To address this challenge, Gong implemented user and item clustering using an inter-user clustering technique based on user ratings, resulting in more personalized recommendation strategies. By identifying cluster groups similar to the target user through item similarities, scalability and sparsity issues inherent in traditional Collaborative Filtering approaches were effectively resolved.

Clustering plays a vital role in Collaborative Filtering recommendation models and is extensively studied across various industries such as tourism, education, and e-commerce. Its ability to efficiently group similar data points contributes significantly to enhancing recommendation accuracy and personalization in diverse application domains. [2]

1.3.5 Neural Network

Neural networks have seen widespread adoption in various domains like speech recognition, image processing, and language translation in recent years. While their utilization in recommendation systems is relatively less compared to other fields, there is a growing interest in incorporating neural networks into recommendation system research. Researchers are exploring neural networks to gather additional insights in scenarios where understanding user preferences solely through historical data is challenging.

For instance, He et al. employed a deep neural network (DNN) to model noisy implicit feedback data, aiming to enhance recommendation system performance. Deep learning techniques offer the potential to boost the effectiveness of recommendation systems significantly. Neural networks are increasingly utilized in recommendation system development to address issues like data sparsity and cold start problems in Collaborative Filtering. They play a crucial role in enhancing system performance by supplementing data and improving recommendation accuracy through advanced modeling techniques. [2]

1.4 Research Trends of Recommendation System Techniques

Figures 1.5 and 1.6 are visualizations that can be used to analyze the research trends of techniques used in the recommendation system field. Figure 1.5 visualizes the number of recommended techniques used in the papers analyzed according to the survey collection criteria set in this paper. Furthermore, Figure 1.6 visualizes the utilization of each recommended technique according to the flow during the year.

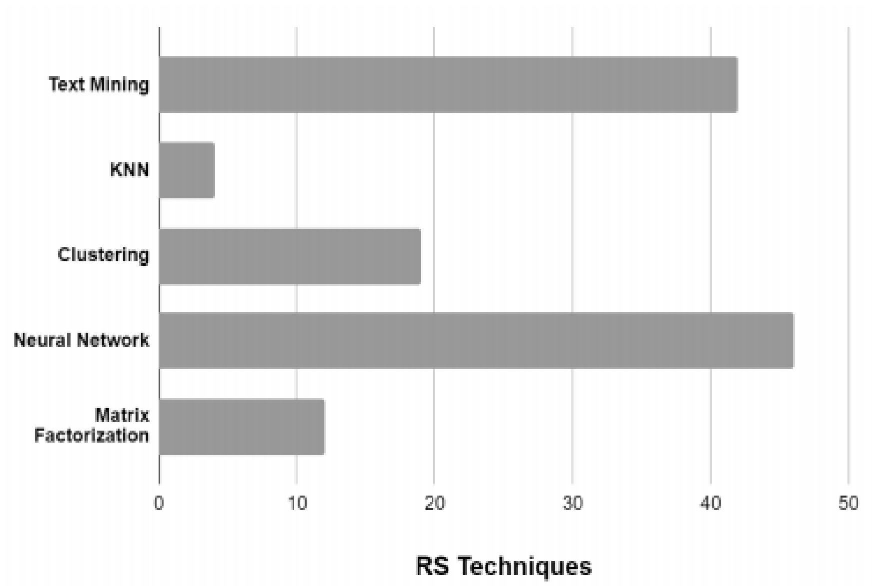


Figure 1.5: Frequency (number) of papers using recommendation techniques during the period investigated in this paper (2010–2020). [2]

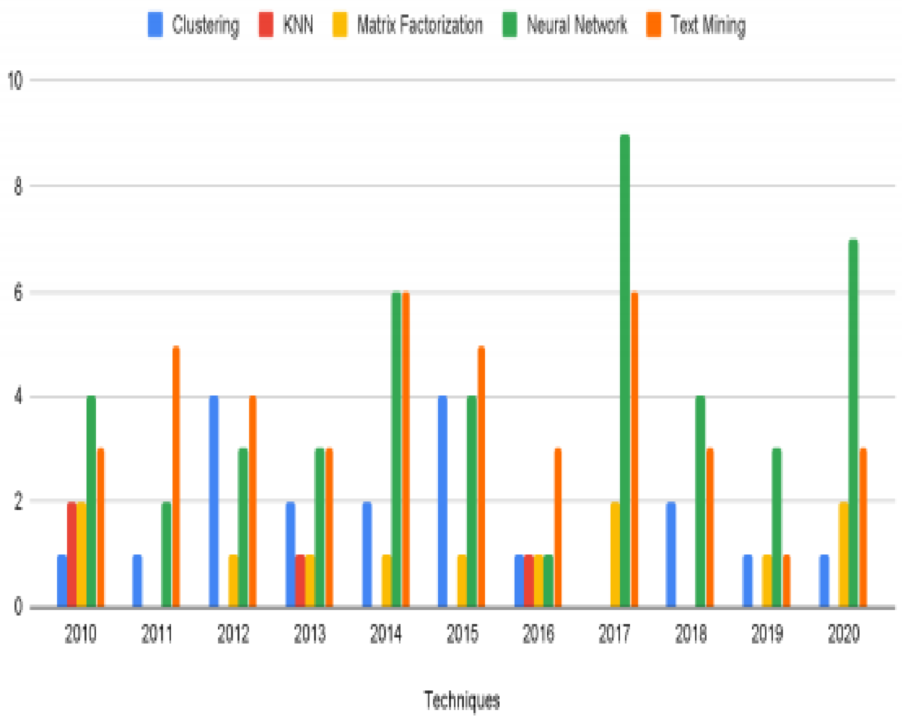


Figure 1.6: Trend in recommendation technique papers by year (by period) during the period investigated in this paper (2010–2020). [2]

Text Mining is an essential technology for analyzing user-selected item characteristics in Collaborative Filtering models like the Item-Based Collaborative Filtering, Content-Based Filtering Recommendation, and Hybrid Recommendation models. This technique finds application across various recommendation system models and remains crucial in sectors such as healthcare, academia, and tourism, which involve substantial text data. Figures 1.5 and 1.6 highlight the active and continuous utilization of text mining in recommendation system research.

Conversely, K-Nearest Neighbor (KNN) has shown limited usage in recommendation system research post-2010 due to challenges like inefficient K value selection, bias issues, and limitations with large data sizes. Clustering is primarily employed to identify user groups akin to Collaborative Filtering users, especially in analyzing location-based data within the travel sector. While clustering remains valuable for evaluating similar groups or items, its frequency as a recommendation technique is decreasing in apps and web services where users provide feedback through 'likes', star ratings, or numerical data.

Matrix Factorization (MF) technology seeks factors expressing user preferences for service-provided items, enabling analysis of diverse data sources beyond numerical item data. The reduced calculation time in MF, achieved through matrix decomposition of user-item evaluations, has led to increased research focus on MF over KNN due to its problem-solving capabilities. However, as businesses adopt large-scale servers, the demand for advanced recommendation system technologies surpassing MF to deliver faster and more accurate results grows.

The advent of smartphones, wearables, and social networking platforms has enabled the collection of various user-related data, including body data, posts, keywords, and images shared on social media. Neural Network technology is increasingly applied in recommendation system research to analyze diverse data types effectively. Particularly, Neural Networks specializing in image analysis and prediction enhance the accuracy and speed of analyzing user-uploaded images or purchased items. This technology can learn user preferences and recommend travel destinations based on various features, surpassing traditional clustering methods. Figures 1.5 and 1.6 underscore the widespread adoption of Neural Network technology as the most commonly used technology in recommendation system research over the past decade. [2]

1.5 Recommendation system based on deep learning methods: a systematic review and new directions

In today's digital landscape, recommender systems (RS) play a vital role in addressing the information overload challenge across diverse sectors like e-commerce, entertainment, and social media. While traditional RS methods have shown success in offering item recommendations, they encounter issues such as cold start problems and data sparsity. The emergence of deep learning has revolutionized various fields like Natural Language Processing (NLP) and image processing, prompting researchers to explore deep learning techniques to enhance RS performance.

Despite the growing interest in deep learning-based RS, there is a scarcity of comprehensive secondary studies in this domain. This study aims to fill this gap by conducting a systematic literature review (SLR) on deep learning-based RSs. By following established SLR guidelines, particularly those outlined by Kitchemen-ham, this paper provides a detailed analysis of top-quality research publications in the field. Through rigorous selection criteria and quality assessment, a set of relevant publications were identified for review.

The findings of this review highlight that autoencoder (AE) models are the most commonly utilized deep learning architectures in RS, closely followed by Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs). Popular datasets for evaluating deep learning-based RS include MovieLens and Amazon review datasets. The study reveals that movie and e-commerce domains are prevalent areas for RS applications. Evaluation metrics such as precision and Root Mean Squared Error are commonly employed to assess the performance of deep learning-based RSs. This SLR provides valuable insights into current trends and challenges in the field, serving as a guide for researchers and practitioners seeking to leverage deep learning for enhancing recommendation systems. [11]

1.6 Social Implications

The social implications of recommender systems present two notable incentive issues. Firstly, once a user establishes their interests profile, there's a risk of free-riding by solely relying on evaluations provided by others. Even if evaluations are gathered implicitly or from user behavior monitoring, this problem persists. Future systems might need to incentivize recommendation contributions by making it a requirement for receiving recommendations, or by offering monetary compensation. Secondly, the open nature of recommendation provision can lead to biased recommendations, with content owners potentially flooding their own materials with positive reviews and disparaging competitors. To counteract this, future systems are likely to introduce measures to deter such manipulative behaviors.

Furthermore, recommender systems raise privacy concerns. While having more information about recommendations aids in evaluation, individuals may be wary of their habits or views

	Contents of recommendation	Explicit entry?	Anonymous?	Explicit entry?	Use of recommendation
GroupLens	a) numeric: 1–5 b) seconds	a) explicit b) monitor reading time	pseudonymous	personalized weighting based on past agreement among recommenders	display alongside articles in existing summary views
fab	numeric: 1–7	explicit	pseudonymous	personalized weighting; combined with content analysis	selection/ filtering
ReferralWeb	mention of a person or a document	mined from public data sources	attributed	assemble referral chain to desired person	display
PHOAKS	mention of a URL	mined from usenet postings	attributed	one person one vote (per URL)	sorted display
Siteseer	mention of a URL	mined from existing bookmark folders	anonymous	frequency of mention in overlapping folders	display

Table 1.1: The technical design space. [4]

being exposed. While some systems allow for anonymous or pseudonymous participation, this doesn't fully address the issue, as some users may desire a balance between privacy and recognition for their contributions.

Similar challenges are observed in academia's peer review system. Editors know that authors with articles under consideration provide the most thorough reviews, creating an incentive issue. Privacy concerns are addressed through blind and double-blind refereeing practices. Incorporating similar practices into automated recommendation systems may help mitigate these challenges. [4] The following tables (table 1.1 and table 1.2 and table1.3) provide an explanation of the above:

	Type of items	**How many**	**Lifetime**	**Cost structure**
GroupLens	netnews articles	thousands per day	1–2 weeks	misses unimportant false positive very small cost hits small value
PHOAKS, SiteSeer, Fab	URLs	hundreds per day	2 days– 2 yesars	misses unimportant false positives small cost hits medium value
ReferralWeb	people	a few million reachable online	many years	depends on how referral chain will be used

Table 1.2: The domain space—characteristics of items evaluated [4]

	Recommenders	**Density of recommendations**	**Consumers**	**Comsumer Taste variability**
GroupLens	subscribers	somewhat dense within newsgroup	Subscribers	high for some newsgroups
Fab	All subscribers	somewhat dense among people served by same collector agent	Subscribers	unknown
ReferralWeb	All authors of online documents	reflects density of underlying social network	Any Web user	unknown
PHOAKS	All usenet authors	extremely sparse	Any Web user	unknown
Siteseer	All subscribers	sparse	Subscribers	high

Table 1.3: The domain space—characteristics of the participants and the set of evaluations [4]

1.7 related work

Title	Authors	Year	Machine Learning Techniques	Dataset	Key Findings
A Hybrid Travel Recommendation System Using Machine Learning	Smith et al.	2018	Collaborative Filtering, K-Means	TripAdvisor, Yelp	Combined collaborative filtering with clustering to improve recommendation accuracy.
Personalized Travel Recommendation System Based on User Reviews	Johnson & Wang	2019	Sentiment Analysis, NLP	TripAdvisor, Expedia	Used sentiment analysis on user reviews to provide personalized recommendations.
Travel Recommendation Using Deep Learning	Lee et al.	2020	Deep Neural Networks	Custom dataset (web scrapped)	Leveraged deep learning to analyze user preferences and predict destinations.
Context-Aware Travel Recommendation System	Patel & Sharma	2021	Contextual Bandits, Decision Trees	Booking.com, Airbnb	Implemented context-aware models to enhance recommendation relevance.
Multi-Modal Travel Recommendation with Data Fusion	Chen & Zhang	2021	Multi-Modal Learning, SVM	Various online travel platforms	Fused data from multiple sources to improve the recommendation system's performance

Table 1.4: table provides a comprehensive overview of various approaches and techniques used in developing travel recommendation systems based on machine learning.

Compared to similar works and previous studies, we can say that our web application is the first application in Algeria that relies on a travel recommendation system and machine learning techniques, as it provides the user with new experiences through their previous desires and opinions and through estimating their budgets, i.e. an accurate study of all their previous desires until the recommendation is with results that they are satisfied with.

1.8 conclustion

In conclusion, the chapter on recommendation techniques elucidates the diverse methodologies employed to enhance recommendation systems. Text mining emerges as a pivotal technology, facilitating semantic analysis and enabling accurate item recommendations. While traditional frequency-based analysis has its limitations, the integration of ontology and fuzzy linguistic modeling enriches text understanding, particularly in scenarios with unclear user preferences or sparse data.

K-Nearest Neighbor (KNN) algorithms, while effective in predicting user preferences, face challenges related to optimal parameter selection and scalability, especially with large datasets. Clustering techniques offer a solution to categorize data efficiently, enhancing recommendation accuracy by grouping similar items or users. Matrix Factorization (MF) technology provides an alternative approach, categorizing data into distinct clusters based on user-item evaluations, thus overcoming scalability issues inherent in KNN.

The advent of Neural Networks introduces advanced modeling techniques to address data sparsity and cold start problems, significantly improving recommendation accuracy. Deep learning architectures like autoencoders (AEs), Convolutional Neural Networks (CNNs), and Recurrent Neural Networks (RNNs) are increasingly adopted, particularly in domains like movie and e-commerce, supported by popular datasets such as MovieLens and Amazon reviews.

Despite the efficacy of these techniques, social implications regarding user privacy and incentivizing recommendation contributions warrant attention. Future recommendation systems may need to implement measures to address biases, manipulative behaviors, and privacy concerns effectively. By integrating practices like blind refereeing and pseudonymous participation, recommendation systems can navigate these challenges while ensuring user trust and system integrity.

In essence, the chapter underscores the dynamic landscape of recommendation system research, driven by innovations in text mining, clustering, matrix factorization, and neural networks. By addressing technical challenges and social implications, recommendation systems can continue to evolve, offering personalized and context-aware recommendations across various domains and industries.

2

Machine Learning and Deep Learning

2.1 Introduction

Artificial intelligence (AI) is a field of study dedicated to developing intelligent machines. Machine learning, a prominent subfield of AI, focuses on enhancing computer program performance through learning from experience. This approach allows computers to execute tasks by leveraging past experiences rather than relying solely on explicit programming. Various sectors like commerce, biology, medicine, engineering, and cyber-security leverage machine learning to gain valuable insights into specific tasks. Notably, Google's search engine is a prime example of a service that extensively employs machine learning. Google tracks user interactions to enhance search result relevance and advertising precision. Additionally, user queries are analyzed over time to reveal public interests. Google Trends, for instance, offers insights into global search trends, aiding entrepreneurs in identifying business opportunities and economists in predicting market trends. This service also proves beneficial in epidemiological studies by aggregating search terms that serve as early indicators of diseases, as demonstrated by Ginsberg et al. (2008) in detecting influenza outbreaks.

The primary objective of machine learning is to make accurate predictions about various phenomena, although prediction remains a challenging task, as famously quoted, "It is difficult to make predictions, especially about the future." Despite the inherent difficulty, prediction plays a crucial role in guiding decisions and behaviors. Platforms like YouTube, owned by Google, utilize machine learning to analyze user preferences and recommend relevant content effectively. Similarly, social networks such as Facebook and LinkedIn employ predictive algorithms to suggest potential connections. Most machine learning techniques rely on inductive learning, where models are developed by generalizing from training examples, assuming applicability to unseen data in the future.

In the past, data collection was a critical aspect of data analysis, with analysts selecting variables based on domain knowledge and limited resources. However, with the advent of the big data era, data accumulation has become more accessible and cost-effective. The exponential growth of stored information, estimated to double every twenty months, presents challenges in effectively utilizing this vast amount of data for meaningful insights and decision-making. [12]

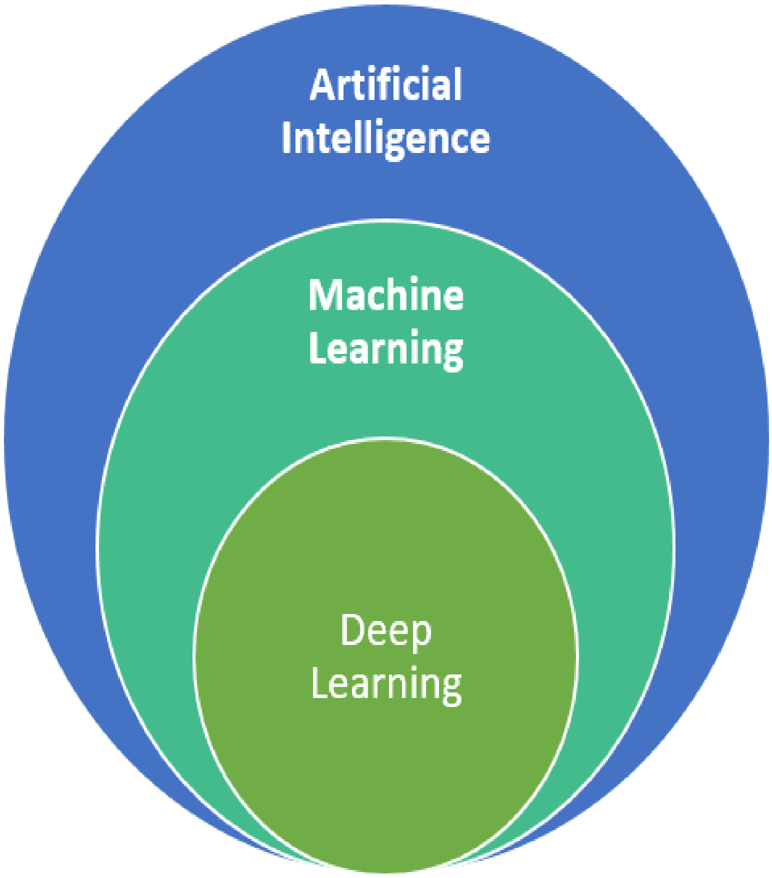


Figure 2.1: artificial intelligence , machine learning and deep learning [2]

2.2 What is Artificial Intelligence?

One intriguing aspect of artificial intelligence (AI) lies in the challenge of defining its precise nature. The issue is twofold: first, understanding what makes something ”artificial” in AI, and second, grasping the concept of intelligence itself. The term ”artificial” implies creation by human design rather than natural processes, distinguishing artificially intelligent entities from naturally intelligent ones. These artificial intelligences, often considered machines, derive their intelligence from human ingenuity, posing questions about the unique properties they possess as products of deliberate construction.

However, defining intelligence itself proves elusive. Webster’s New World Dictionary offers various definitions, encompassing learning, problem-solving, and adaptability. Yet, applying these definitions to inanimate machines raises questions about whether they can truly exhibit intelligence or if such attribution is a matter of arbitrary definition.

The crux of the issue lies in discerning the essential features of intelligence, a task akin to establishing a theory of intelligence in the world. It requires careful consideration of what qualifies as intelligent and what does not. For instance, while human beings are commonly regarded as intelligent, the presence of emotions like anger and jealousy complicates the matter.

If these emotions are deemed integral to intelligence, then machines capable of emulating them might also be considered intelligent.

Ultimately, determining the boundaries of intelligence involves identifying its core features through examination of exemplars and considering whether machines can possess these traits. Yet, even if human beings are taken as benchmarks of intelligence, the presence of complex emotions challenges clear delineation. Thus, while definitions aid in understanding human intelligence, their application to machines remains fraught with complexity. [13]

2.3 Machine Learning

In 1959, Arthur Samuel defined machine learning as “a field of study that gives computers the ability to learn without being explicitly programmed.”Sometimes it is difficult to fulfil some kinds of task by giving explicit programming instructions. In this case, the best approach is to enable computers to have the ability to learn from the data.

2.3.1 definition ”1”

Machine Learning is a field of study to give computer systems the ability to “learn”with data, without being explicitly programmed.

Machine learning is ideal for exploiting the opportunities hidden in big data [3]

Figure 2.2: The motivation of machine learning. [3]

2.3.2 definition ”2”

Machine learning represents a dynamic field of computational algorithms aimed at mimicking human intelligence through learning from the environment. It serves as a cornerstone in the era of big data, offering solutions across various domains such as pattern recognition, computer vision, finance, and biomedical applications. In the realm of healthcare, particularly in cancer treatment, machine learning holds significant promise. With over half of cancer patients undergoing radiotherapy, a complex treatment modality, there arises a need for optimizing and automating its processes. These processes, spanning from initial consultation to treatment response monitoring, involve intricate human-machine interactions and decision-making stages. Incorporating machine learning algorithms into tasks such as quality assurance, treatment planning, and outcome prediction could enhance the safety and efficacy of radiotherapy, ultimately improving patient outcomes. The adaptability of machine learning algorithms to new contexts enables advancements in radiotherapy practice, ensuring better patient care. [14]

2.3.3 Types of Machine learning

Machine learning algorithms can be mainly divided into supervised learning, unsupervised learning and reinforcement learning : [3]

2.3.3.1 Supervised Learning

How it works:

Supervised Learningaims to learn from a labelled training set suchthat the resulting model can be effectively applied to unseen data. Start-ing from the analysis of a given training dataset, which is a collectionof input–output pairs, the supervised learning algorithm produces aninferred function to make predictions about the output values. [3]

The process:

The computer is fed labeled data. A data scientist "supervises" the process by confirming the computer's accurate responses and correcting incorrect responses. [15]

Example:

A programmer is trying to "teach" a computer how to tell the difference between fish and birds.

The engineer feeds the computer model a set of labeled data. In this case, it's clearly identifiable images of fish and birds.

Over time, the model starts recognizing patterns. For instance, it may recognize that birds have wings and beaks and fish have scales and gills. Once the model has sufficient data, the engineer begins feeding it unlabeled data in the form of unidentified photos.

The engineer then tests the model to see if it can accurately distinguish between birds and fish. [15]

2.3.3.2 Unsupervised Learning studies

How it works:

Unsupervised Learning studies how systems can infer a function to describe a hidden structure from unlabeled data. [3]

The process:

The computer is fed unlabeled data. The model processes the data and looks for patterns. [15]

Example:

Picture this: You own a retail store and you're developing a sales and marketing plan to entice unique customer segments to shop with you before the winter holidays. The only problem is you don't know your target audience yet.

To figure it out, you use unsupervised learning and feed it unlabeled input data about your customers' demographics, purchasing behaviors and history, and location. It's important the data is unlabeled so the computer can classify customer segments on its own.

Using the data, the computer autonomously searches for patterns and extracts key features.During this process, the algorithm will identify patterns and group shopper data based on similarities and differences in processes called "clustering" and "segmentation."

Finally, you will receive an output with final customer segments that might include "last-minute shoppers," "parents of young children," "early tech adopters," or "budget shoppers." With a better understanding of the various customer segments, you can develop hyper-targeted marketing and sales strategies. [15]

2.3.3.3 Reinforcement Learning(RL)

How it works:

Reinforcement Learning(RL) is a learning method to identify optimal actions that interacts with its environment by maximizing the cor-responding rewards. This method allows machines and software agents to automatically determine the ideal behavior within a specific context based on a feedback loop from the environment. [3]

The process:

The model is fed training data. To fine-tune the model, an engineer "rewards" the computer's correct answers and "punishes" inaccurate outputs. When the model answers incorrectly, the engineer feeds it the correct answer. [15]

Example:

Reinforcement learning is used to help machines master complex tasks that come with massive data sets, such as driving a car. For instance, a vehicle manufacturer uses reinforcement learning to teach a model to keep a car in its lane, detect a possible collision, pull over for emergency vehicles, and stop at red lights. [15]

Through lots of trial and error, the program learns how to make decisions that keep the car on the road and passengers safe.

Figure 2.3 provides a summary of the main categories of machine learningand corresponding examples of financial applications.In contrast to supervised learning, unsupervised learning algorithms areused when the data is neither classified nor labeled. As compared to super-vised learning and unsupervised learning, RL is different in terms of goals.While the goal in unsupervised learning is to find similarities and differencesbetween data points, the goal in reinforcement learning is to find a suit-able action model that would maximize the reward of the agent. [3]

type	Description
Supervised learning	Supervised learning requires a training dataset that covers examples for the input as well as labeled answers or target values for the output. An example could be the prediction of active users subscribed to a market platform in a month's time as output (considered as the target variable or y variable) based on different input characteristics, such as the number of sold products or positive user reviews (often referred to as input features or x variables). The pairs of input and output data in the training set are then used to calibrate the open parameters of the ML model. Once the model has been successfully trained, it can be used to predict the target variable y given new or unseen data points of the input features x. Regarding the type of supervised learning, we can further distinguish between regression problems, where a numeric value is predicted (e.g., number of users), and classification problems, where the prediction result is a categorical class affiliation such as "lookers" or "buyers".
Unsupervised learning	Unsupervised learning takes place when the learning system is supposed to detect patterns without any pre-existing labels or specifications. Thus, training data only consists of variables x with the goal of finding structural information of interest, such as groups of elements that share common properties (known as clustering) or data representations that are projected from a high-dimensional space into a lower one (known as dimensionality reduction) (Bishop 2006). A prominent example of unsupervised learning in electronic markets is applying clustering techniques to group customers or markets into segments for the purpose of a more target-group specific communication.
Reinforcement learning	In a reinforcement learning system, instead of providing input and output pairs, we describe the current state of the system, specify a goal, provide a list of allowable actions and their environmental constraints for their outcomes, and let the ML model experience the process of achieving the goal by itself using the principle of trial and error to maximize a reward. Reinforcement learning models have been applied with great success in closed world environments such as games (Silver et al. 2018), but they are also relevant for multi-agent systems such as electronic markets (Peters et al. 2013).

Table 2.1: Overview of types of machine learning [5]

2.4 Deep Learning

2.4.1 Historical Trends in Deep Learning

It is easiest to understand deep learning with some historical context. Rather than providing a detailed history of deep learning, we identify a few key trends:

• Deep learning has had a long and rich history, but has gone by many names, reflecting different philosophical viewpoints, and has waxed and waned in popularity.

• Deep learning has become more useful as the amount of available training data has increased.

• Deep learning models have grown in size over time as computer infrastructure (both hardware and software) for deep learning has improved.

Deep learning has solved increasingly complicated applications with increasing accuracy over time. [16]

2.4.2 Why Is Deep Learning So Successful?

In any data-driven endeavor, success hinges on identifying the right metrics and devising effective measurement strategies. This underscores the crucial role of feature selection and design in machine learning. These tasks demand domain expertise, statistical analysis, and iterative experimentation with various feature combinations, often consuming a substantial portion of time and resources, sometimes up to 80% of the total project budget (Kelleher and Tierney, 2018). Feature design, particularly, stands out as an area where deep learning excels over traditional methods. Unlike traditional approaches that rely heavily on manual feature engineering, deep learning adopts a more autonomous approach by learning feature representations directly from raw data.

Consider the example of body mass index (BMI), a derived feature calculated from weight and height measurements, commonly used in medical contexts to classify individuals based on weight status. This illustrates how derived features can provide valuable insights beyond raw data inputs, exemplifying the efficacy of handcrafted features, such as BMI, conceptualized as far back as the eighteenth century by Adolphe Quetelet.

In machine learning projects, a significant portion of time and effort is dedicated to identifying or crafting features tailored to the specific task at hand. Deep learning offers a distinct advantage by autonomously learning useful feature representations from data, leveraging its capability to handle large datasets effectively. Consequently, deep learning models often outperform their counterparts reliant on manually engineered features, especially in domains characterized by high-dimensional datasets, such as image processing, where each pixel constitutes a feature. Hand-engineering features in such complex domains, like face recognition or machine translation, poses formidable challenges, further highlighting the superiority of deep learning methodologies. [17]

2.4.3 Definitions of Deep Learning

2.4.3.1 Definitions”1”

“A sub-field within machine learning that is based on algorithms for learning multiple levels of representation in order to model complex relationships among data. Higher-level features and concepts are thus defined in terms of lower-level ones, and such a hierarchy of features is called a deep architecture. Most of these models are based on unsupervised learning of representations.” (Wikipedia on “Deep Learning” around March 2012.) [18]

2.4.3.2 Definitions”2”

“Deep learning is a set of algorithms in machine learning that attempt to learn in multiple levels, corresponding to different levels of abstraction. It typically uses artificial neural networks. The levels in these learned statistical models correspond to distinct levels of concepts, where higher-level concepts are defined from lower-level ones, and the same lowerlevel concepts can help to define many higher-level concepts.” “Deep Learning” as of this most recent update in October 2013 [18]

2.4.4 Overview of deep learning architectures

During automated model building, the input is used by a learning algorithm to identify patterns and relationships that are relevant for the respective learning task. As described above, shallow ML requires well-designed features for this task. On this basis, each family of learning algorithms applies different mechanisms for analytical model building. For example, when building a classification model, decision tree algorithms exploit the features space by incrementally splitting data records into increasingly homogenous partitions following a hierarchical, tree-like structure. A support vector machine (SVM) seeks to construct a discriminatory hyperplane between data points of different classes where the input data is often projected into a higher-dimensional feature space for better separability. These examples demonstrate that there are different ways of analytical model building, each of them with individual advantages and disadvantages depending on the input data and the derived features (Kotsiantis et al. 2006).

By contrast, DL can directly operate on high-dimensional raw input data to perform the task of model building with its capability of automated feature learning. Therefore, DL architectures are often organized as end-to-end systems combining both aspects in one pipeline. However, DL can also be applied only for extracting a feature representation, which is subsequently fed into other learning subsystems to exploit the strengths of competing ML algorithms, such as decision trees or SVMs.

Various DL architectures have emerged over time (Leijnen and van Veen 2020; Pouyanfar et al. 2019; Young et al. 2018). Although basically every architecture can be used for every task, some architectures are more suited for specific data such as time series or images. Architectural variants are mostly characterized by the types of layers, neural units, and connections they use. Table 2.1 summarizes the five groups of convolutional neural networks (CNNs), recurrent neural networks (RNNs), distributed representations, autoencoders, and generative adversarial neural networks (GANs). They provide promising applications in the field of electronic markets.

Table 2.2: Overview of deep learning architectures [5]

Architecture	Description
Convolutional neural network(CNN)	CNNs are mainly applied for tasks related to computer vision and speech recognition. They are able to address tasks involving datasets with spatial relationships, where the columns and rows are not interchangeable (e.g., image data). Their network architecture comprises a series of stages that allow hierarchical feature learning as determined by the respective modeling task. For example, when considering object recognition in images, the first few layers of the network are responsible for extracting basic features in the form of edges and corners. These are then incrementally aggregated into more complex features in the last few layers resembling the actual objects of interest, such as animals, houses, or cars. Subsequently, the auto-generated features are used for prediction purposes to recognize objects of interest in new images (Goodfellow et al. 2016).
Recurrent neural network (RNN)	RNNs are designed explicitly for sequential data structures such as time-series data, event sequences, and natural language. Their architecture offers internal feedback loops and therefore enables sequential pattern learning to model time dependencies by forming a memory. Simple RNN architectures are problematic since they suffer from vanishing gradients, resulting in little or no influence of early memories. More sophisticated architectures, such as long short-term memory (LSTM) networks with advanced attention mechanisms, attend to this problem. RNNs are typically applied for time series forecasting, predicting process behavior (Heinrich et al. 2021), and NLP tasks such as sequence transduction and neural machine translation (LeCun et al. 2015).
Distributed representation	Distributed representations play an essential role in feature learning and language modeling in NLP tasks, where language entities such as words, phrases, and sentences are projected into numerical representations within a unified semantic space in the form of embeddings. Word embeddings, for example, encode discrete words into dense feature vectors with low dimensionality. Thus, in contrast to classic text representation models, such as one-hot encodings and bag-of-words (BoW), word embeddings overcome the problem of sparse encodings while preserving semantic relationships between words. This means that words, which occur in similar contexts in a corpus, are also closely positioned to each other in the vector space. On this basis, advanced language models can be developed to perform challenging downstream tasks, such as question-answering, sentiment analysis, and named entity recognition (Liu et al. 2020). Distributed representations are often applied in combination with RNNs to perform tasks with sequential dependencies.

Table 2.2 – Continued from previous page

Architecture	Description
Autoencoder	Autoencoders work similarly to word embeddings since they provide a dense feature representation of the input data. However, they are not limited to natural language data but can be applied to any type of input. Such architectures usually consist of an encoding stage where the input is compressed into a low-dimensional representation and a decoding stage in which the network tries to reconstruct the original input from the learned features. In this way, the network is forced to keep meaningful information in the latent representation while disregarding irrelevant noise (Goodfellow et al. 2016). Autoencoders are commonly applied for unsupervised feature learning and dimensionality reduction in combination with other subsequent learning systems. However, due to their capability of quantifying reconstruction errors, which are assumed to be significantly higher for anomalous samples than for regular instances, they can also be applied for detecting anomalies, such as fraudulent activities in financial markets (Paula et al. 2016).
Generative adversarial neural network (GAN)	Generative adversarial neural networks belong to the family of generative models that aim at learning a probability distribution over a set of training data so that the network can randomly generate new data samples with some variation. For this purpose, GANs consist of two competing sub-networks. The first network is a generator network that captures the distribution of the input and generates new examples. The second network is a discriminator network trying to distinguish real examples from artificially generated ones. Both networks are trained together in a non-cooperative zero-sum game where one network's gain is another one's loss until the discriminator can no longer distinguish between both types of samples. On this basis, GANs are likely to revolutionize domains in which continuously new content or novel product configurations are created (e.g., the composition of art and music, design of fashion), or where content is converted from one representation to another (e.g., text to image for product descriptions) (Pan et al. 2019). At the same time, however, such approaches also pose severe threats with societal implications when abusing them for malicious purposes. In particular, the generation of "deepfake" content in the form of abusive speeches and misleading news to manipulate public opinions or distort financial markets is concerning (Westerlund 2019).

2.5 Artificial Intelligence [AI] Vs Machine Learning [ML] Vs Deep Learning [DL]

Machine learning and deep learning have clear definitions, whereas what we consider AI changes over time. For instance, optical character recognition used to be considered AI, but it no longer is. However, a deep learning algorithm trained on thousands of handwritings that can convert those to text would be considered AI by today's definition.

Machine learning and deep learning power various applications, including those performing natural language processing, image recognition and classification. These technologies help enterprises augment their workforce by having intelligent machines tackle mundane, repetitive tasks. This frees up employees to focus on creative or high-thinking jobs

In the following table2.3 we have explained the most important differences between them:

	Artificial Intelligence	**Machine Learning**	**Deep Learning**
optimal data volumes	varying data volumes	thousands of data points	big data :millions of data pionts
Outputs	Anything from predictions to recommendations to decision-making	Numerical value,like a classification or score	Anything from numerical values to free-form elements , like free text sound
How it Works	Machines are programmed to mimic human activity with human-like accuracy	Uses various types of automated algorithms that learn to model functions and predict future actions from data	Uses neural networks that pass data through many processing layers to interpret data featurs and relationships
How it's managed	Algorithms require human oversight in order to function properly	Algorithms are directed by data analysts to examine specific variables in data sets	Algorithms are largely self direcred on data analysis once they're put into production

Table 2.3: Artificial Intelligence [AI] Vs Machine Learning [ML] Vs Deep Learning [DL] [6]

2.6 conclustion

In conclusion, the exploration of machine learning and deep learning techniques within the context of recommendation systems reveals a dynamic landscape of evolving methodologies and applications. Across various subdomains such as supervised learning, unsupervised learning, reinforcement learning, and deep learning, each technique offers unique strengths and applications.

Supervised learning thrives on labeled data, allowing models to predict outcomes or classify inputs based on established patterns. Unsupervised learning, on the other hand, delves into the realm of pattern detection without predefined labels, offering insights into data structures and relationships. Reinforcement learning introduces the concept of an agent interacting with an environment to maximize rewards, making it particularly suitable for dynamic decision-making scenarios.

Deep learning, with its hierarchical representation learning and automated feature extraction, stands out as a transformative force in the field. Its ability to process high-dimensional data directly, without the need for manual feature engineering, has propelled advancements in domains ranging from image recognition to natural language processing.

The historical trends in deep learning underscore its gradual ascent fueled by the availability of large datasets and advancements in computational infrastructure. Its success lies in its autonomous feature learning capabilities, which alleviate the burden of manual feature engineering and enable deeper insights into complex datasets.

Furthermore, the distinction between artificial intelligence, machine learning, and deep learning clarifies their roles in driving technological innovation. While AI encompasses a broad spectrum of intelligent behaviors, machine learning and deep learning represent specialized subsets focused on automated learning from data.

In summary, the intersection of machine learning and deep learning with recommendation systems presents a fertile ground for innovation, offering opportunities to enhance user experiences, personalize recommendations, and tackle real-world challenges across diverse domains. As research and development in these areas continue to evolve, the future holds promise for increasingly intelligent and adaptive recommendation systems tailored to individual preferences and contexts.

3
Conception

3.1 Introduction

Before traveling, usually someone will make a plan in advance about the location to be visited and the time of departure. This is done to avoid problems, one of which is the distance to be traveled and the time needed does not match expectations permission In this chapter, we'll present the solution, then go into more detail. We are going to describe The travel recommendation system and the most important features that the site offers to the user. Including what is required UML diagrams.

3.2 General Architecture of System

This section describes the general structure of the project, which The system consists of two main components: Frontend , systems(web app ,recommendation system), and a database. These components are deployed on the cloud infrastructure and enable the storage of all transferred data. Figure 3.1 illustrate the detailed architecture.

1. Frontend :

   It consists of two sections, each of which has a role in the site:

   - **users:**

     The user will have a multi-use interface on the site, and his use will be limited to his needs Users will interact with the front end of the app by performing actions such as searching for destinations, viewing details, making reservations, etc.

   - **Administrator:**

     The administrator shall have unlimited and extended service It interacts with the backend to manage data processing and model training

2. systems: It consists of two systems

   - **web app:** The system provides a variety of comprehensive services (for hotels, restaurants, and places), flight reservations, and others

   - **system recommendation:** A recommendation system that suggests the user the best places

3. Database :

   The backend interacts with the database to store and retrieve data related to destinations, user profiles, reservations, etc These components are deployed on the cloud infrastructure and enable all storage units and consist of two sections:

   - **Database**

   - **Data Source**

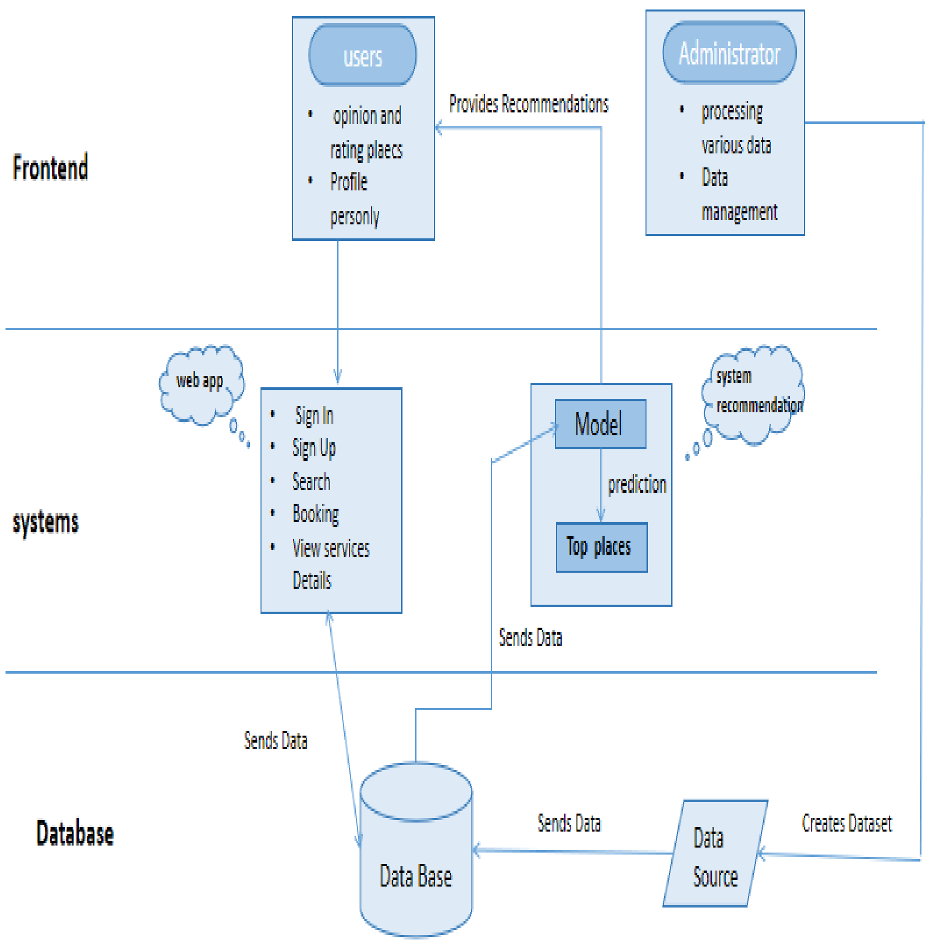


Figure 3.1: The general Architecture Of System

3.3 Detailed Architecture of System

In this section, we provide a detailed description of the system's architecture.

3.3.1 Data Base

3.3.1.1 Data collection

Data collection is the first step of our system. It allows us to obtain data from the real world and Design a model database (educational examples), which will be used in the following steps. we We will collect as much data as possible. In the process of collecting this data set it was difficult Since most of the datasets were multimodal, but we were able to find a free balanced set We can edit in Kaggle platform where this dataset is a dataset containing many tourist attractions in 5 major cities in Indonesia, namely Jakarta, Yogyakarta, Semarang, Bandung, Surabaya. This dataset also consists of 3 files, which are:

- Tourism_with_id.csv which contains information on tourist attractions in 5 major cities in Indonesia 400
- user.csv which contains dummy user data to generate recommendation features based on the user
- tourism_rating.csv contains 3 columns, namely the user, the place, and the rating given, serves to create a recommendation system based on the rating

  In the figure 3.2 and figure 3.3 and figure 3.4 and we show the four files

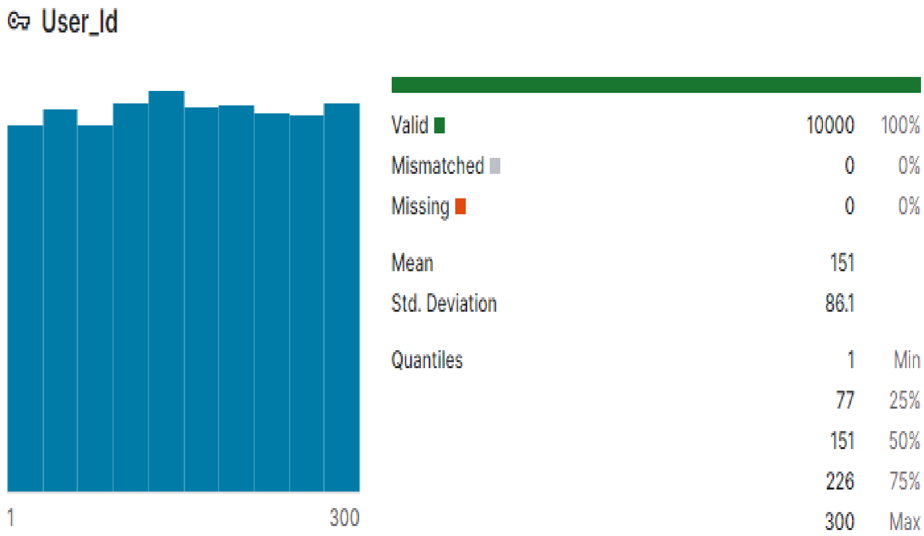


Figure 3.2: The column of tourism rating

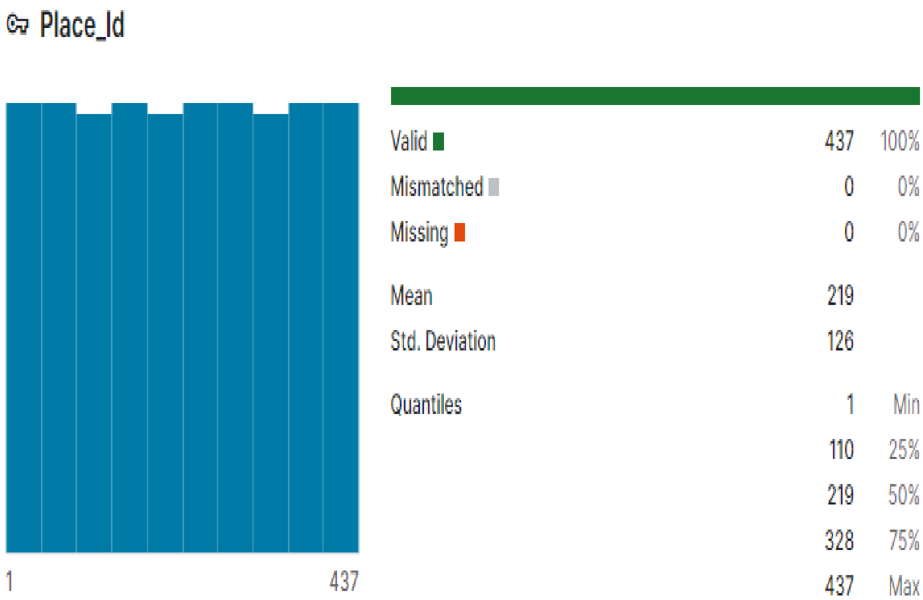


Figure 3.3: The column of tourism with id

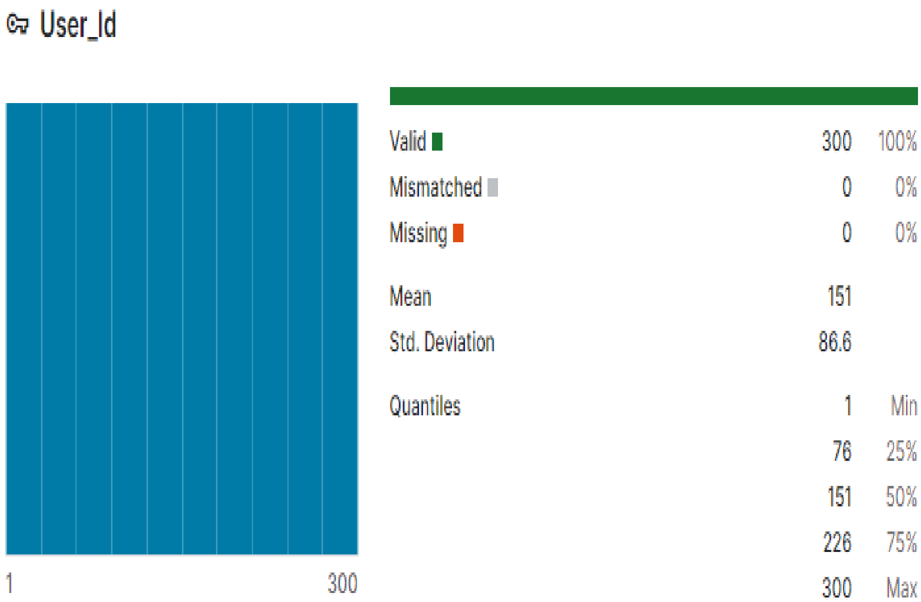


Figure 3.4: The column of user

3.3.1.2 Data Preprocess

After collecting the data, we need to prepare it for further steps. The data is balanced and error-free, to prepare it we can be divided the task into a few general, significant steps:

- **Correlation Matrix:**

  This is an important step in pre-processing machine learning pipelines. Since the correlation matrix is a common tool used to compare the coefficients of correlation between

different features in a dataset. It allows us to identify variables that have high degrees of correlation and allows us to reduce the number of features we may have in a dataset [19]
So what is a correlation coefficient?

Correlation Coefficient

Is a value between -1 and +1 that denotes both the strength and directionality of a relationship between two variables.

- The closer the value is to 1 (or -1), the stronger the relationship.
- The closer a number is to 0, the weaker the relationship.

A negative coefficient will demonstrate that the relationship is negative; that is, as one number increases, the other will decrease. Similarly, a positive coefficient indicates that as one value increases, so does the other [19].

- **Data Normalization** In this step, we will normalize the features. This includes expanding

  the range of features to ensure they fall within them Uniform range. In the absence of normalization, features with larger sizes will have more weight Instinctively, where :

  - Normalizes Place Ratings values between 0 and 1 using minimum and maximum normalization.
  - It divides the data into training and validation sets,
  - We scale the feature distribution such that the mean of the observed values is 0 and the standard deviation is 0 It is 1 with StandardScaler from sklearn.

- **Data Splitting**

  The data splitting method can be implemented once we specify a splitting ratio, such as 80/20, 70/30, 60/40, and even 50/50 are also used in practice. But a commonly used ratio is 80/20

  - It splits the data into training and validation sets, with 80% of the data used for training and 20% for validation.

We will split the dataset into 80% training sets and 20% test sets with the data frame transformed to NumPy array. As shown in this figure: 3.5

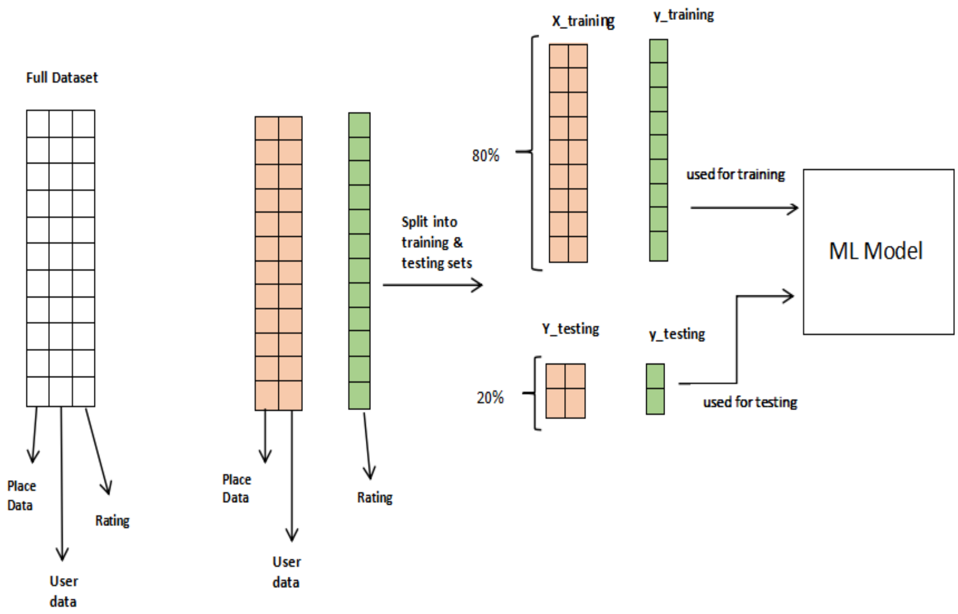


Figure 3.5: Train Test Split

3.3.1.3 Study the most important data variables

- We studied the age group of tourists between 20 and 40 years, as the age group of most tourists was between 20 and 35 years in the database used , shown in the following image 3.6

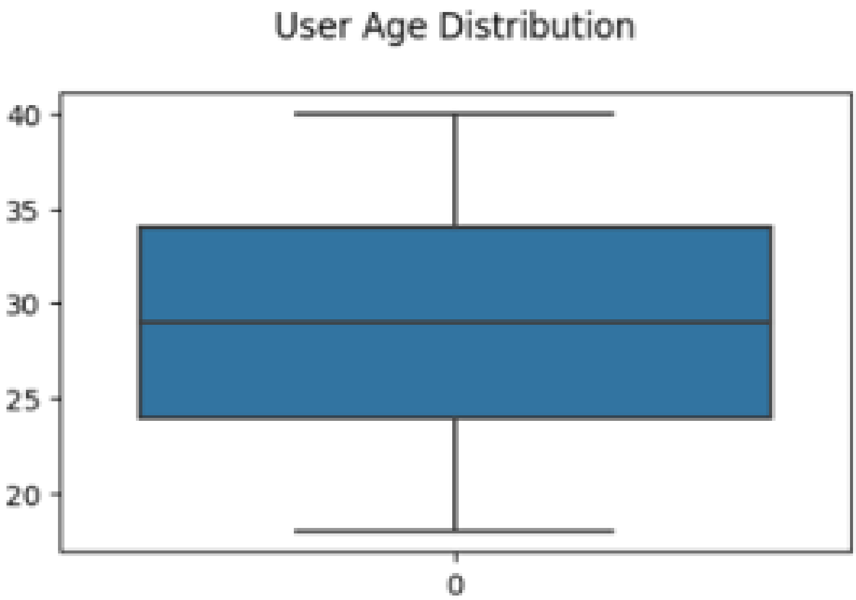


Figure 3.6: User Age Distribution

- Here we studied the distribution of tourists in a specific city at different prices to study the prices and make them suitable for tourists , shown in the following image 3.7

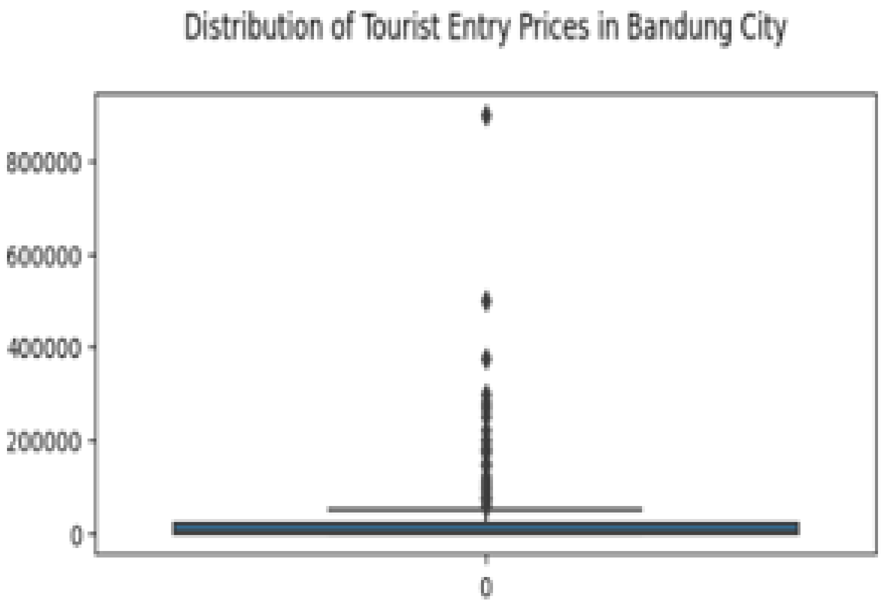


Figure 3.7: Distribution of Tourist Entry Prices in Bandung City

- Here we studied the number of the most rated tourist places , shown in the following image 3.8

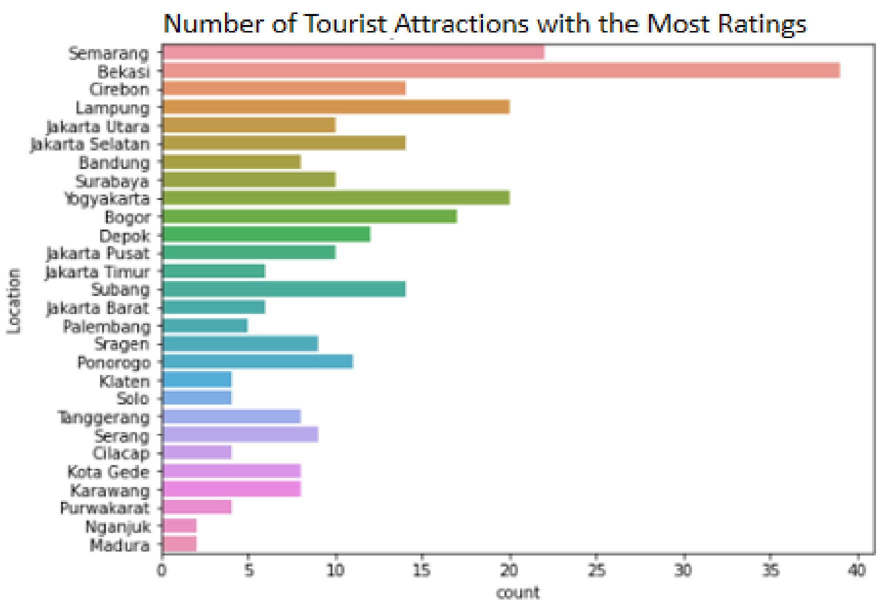


Figure 3.8: Number of the most rated tourist places

3.3.1.4 Types of Data for building recommendation systems

There are two kinds of data available for building a recommendation system. There are [20]:

1. **Explicit feedback:** Explicit feedback is the data about user explicit feedback(ratings etc) about a product. It tells directly that users like a product or not.
2. **Implicit feedback:** In implicit feedback, we don't have the data about how the user rates a product. Examples for implicit feedback are clicks, watched movies, played songs, purchases or assigned tags.

3.3.1.5 Types of feedback in the database-based LightFM

There are different types of feedback that a LightFM can use to improve its recommendations, such as explicit feedback, where the user provides explicit ratings or reviews of items, or implicit feedback, where the system infers the user's preferences based on their actions. [21]

- One type of feedback is explicit feedback, which is input from users regarding their interest in an item. This is the most useful information because it comes directly from the user and shows his direct interest in the item, which is the type used in the recommendation system. As for implicit feedback, it is the information that is produced after observing the users' behavior, and here it is indirect. [22]

When creating the model, I implemented two types of recommendation algorithms, the first is recommenderNET and the second is lightFM. We will explain in detail the two below :

3.3.2 Model Conseption numbre -1- RecommenderNET

In creating this model, I relied on a type of recommendation algorithm called Recommender-NET **how it is work ?**

3.3.2.1 Data Understanding

- The code begins by importing necessary libraries and defining constants like the data path such as NumPy, Pandas, and TensorFlow.
- Load Data The load_data‘ function reads the necessary CSV files containing information about tourism destinations, ratings, and users.

3.3.2.2 Data Preprocessing

- The ‘Preprocess Data’ function prepares data for training. The ‘preprocess_data()‘ function processes the preloaded data, including encoding users and places, normalizing ratings, and obtaining information about the number of users and places, minimum rating, and maximum rating

3.3.2.3 Embedding Layer Class

- The ‘EmbeddingLayer‘ class is defined to create an embedding layer, which maps each user and place to a dense vector space of a specified size.

3.3.2.4 RecommenderNet Model

The ‘RecommenderNet‘ function defines the recommendation model using TensorFlow’s Keras API. It consists of compact layers for users and places, followed by sequential, dropout, and dense layers. The output is a single value representing the expected rating. Below are the details of the model components:

1. **Input Layers:**

   Two layers for entering user and location identifiers.

2. **Embedding Layers:**

   Each user and each place is represented as an embedding vector. These embeddings capture the latent factors of users and places in a low-dimensional space, where similar users and places are close together.

3. **Concatenation:**

   User and location embeddings are concatenated into a single vector, which serves as input for subsequent layers.

4. **Dropout:**

   Dropout layers are used to prevent overfitting by randomly dropping part of the input units during training.

5. **Dense Layers:**

Fully connected dense layers handle sequential embeddings. These layers learn complex patterns in the data.

6. **Activation function:**

   The ReLU activation function is used in the hidden layers to introduce nonlinearity.

7. **Output Layer:**

   A single neuron with a sigmoid activation function is used as the output layer. It produces a value between 0 and 1, representing the predicted rating for the user-place pair.

8. **Loss function:**

   Mean Square Error (MSE) is used as the loss function to measure the difference between predicted and actual ratings.

9. **Optimizer:**

   The Adam optimizer is used to update the model weights based on the calculated gradients.

3.3.2.5 Result

In the end we get the desired result, The model learns to predict ratings for users' place pairs based on patterns in the training data where interactions have not yet occurred. These predicted ratings can then be used to create personalized recommendations for users, suggesting places they are likely to enjoy based on their preferences and past behaviour.

3.3.2.6 Model Evaluation and training

In this recommendation system, the model is evaluated using mean squared error (MSE) and mean absolute error (MAE) metrics. These evaluation metrics provide insights into how well the model is performing in terms of predicting the ratings of tourism destinations.

The model is trained using the training data, and during each epoch of training, the loss, MAE, and MSE are calculated both for the training set and the validation set. The training process aims to minimize the loss function, which in this case is the mean squared error.

After training is complete, the training history, including the loss and validation loss over epochs, is plotted to visualize the training progress and check for overfitting or underfitting.

This evaluation approach allows for assessing the performance of the recommendation system model in predicting tourism destination ratings accurately, providing valuable feedback for model improvement and optimization.

The following images 3.9 and image 3.10 illustrate training the model :

```
Epoch 1/20
250/250 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.3541 - mae: 0.4855 - mse: 0.3541 - val_loss: 0.3540 - val_
mae: 0.4849 - val_mse: 0.3540
Epoch 2/20
250/250 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.3591 - mae: 0.4892 - mse: 0.3591 - val_loss: 0.3540 - val_
mae: 0.4849 - val_mse: 0.3540
Epoch 3/20
250/250 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.3520 - mae: 0.4832 - mse: 0.3520 - val_loss: 0.3540 - val_
mae: 0.4849 - val_mse: 0.3540
Epoch 4/20
250/250 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.3536 - mae: 0.4857 - mse: 0.3536 - val_loss: 0.3540 - val_
mae: 0.4849 - val_mse: 0.3540
Epoch 5/20
250/250 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.3553 - mae: 0.4864 - mse: 0.3553 - val_loss: 0.3540 - val_
mae: 0.4849 - val_mse: 0.3540
Epoch 6/20
250/250 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.3469 - mae: 0.4774 - mse: 0.3469 - val_loss: 0.3540 - val_
mae: 0.4849 - val_mse: 0.3540
Epoch 7/20
250/250 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.3532 - mae: 0.4841 - mse: 0.3532 - val_loss: 0.3540 - val_
mae: 0.4849 - val_mse: 0.3540
Epoch 8/20
250/250 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.3546 - mae: 0.4867 - mse: 0.3546 - val_loss: 0.3540 - val_
mae: 0.4849 - val_mse: 0.3540
Epoch 9/20
250/250 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.3539 - mae: 0.4841 - mse: 0.3539 - val_loss: 0.3540 - val_
mae: 0.4849 - val_mse: 0.3540
Epoch 10/20
250/250 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.3479 - mae: 0.4807 - mse: 0.3479 - val_loss: 0.3540 - val_
mae: 0.4849 - val_mse: 0.3540

Epoch 11/20
250/250 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.3523 - mae: 0.4826 - mse: 0.3523 - val_loss: 0.3540 - val_
mae: 0.4849 - val_mse: 0.3540
Epoch 12/20
250/250 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.3571 - mae: 0.4887 - mse: 0.3571 - val_loss: 0.3540 - val_
mae: 0.4849 - val_mse: 0.3540
Epoch 13/20
250/250 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.3512 - mae: 0.4834 - mse: 0.3512 - val_loss: 0.3540 - val_
mae: 0.4849 - val_mse: 0.3540
Epoch 14/20
250/250 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.3525 - mae: 0.4832 - mse: 0.3525 - val_loss: 0.3540 - val_
mae: 0.4849 - val_mse: 0.3540
Epoch 15/20
250/250 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.3538 - mae: 0.4857 - mse: 0.3538 - val_loss: 0.3540 - val_
mae: 0.4849 - val_mse: 0.3540
Epoch 16/20
250/250 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.3539 - mae: 0.4845 - mse: 0.3539 - val_loss: 0.3540 - val_
mae: 0.4849 - val_mse: 0.3540
Epoch 17/20
250/250 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.3584 - mae: 0.4884 - mse: 0.3584 - val_loss: 0.3540 - val_
mae: 0.4849 - val_mse: 0.3540
Epoch 18/20
250/250 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.3471 - mae: 0.4798 - mse: 0.3471 - val_loss: 0.3540 - val_
mae: 0.4849 - val_mse: 0.3540
Epoch 19/20
250/250 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.3562 - mae: 0.4883 - mse: 0.3562 - val_loss: 0.3540 - val_
mae: 0.4849 - val_mse: 0.3540
Epoch 20/20
250/250 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - loss: 0.3514 - mae: 0.4820 - mse: 0.3514 - val_loss: 0.3540 - val_
mae: 0.4849 - val_mse: 0.3540
```

Figure 3.9: Training the model

In the following image 3.11 is a data chart representing the proportions of the training model:

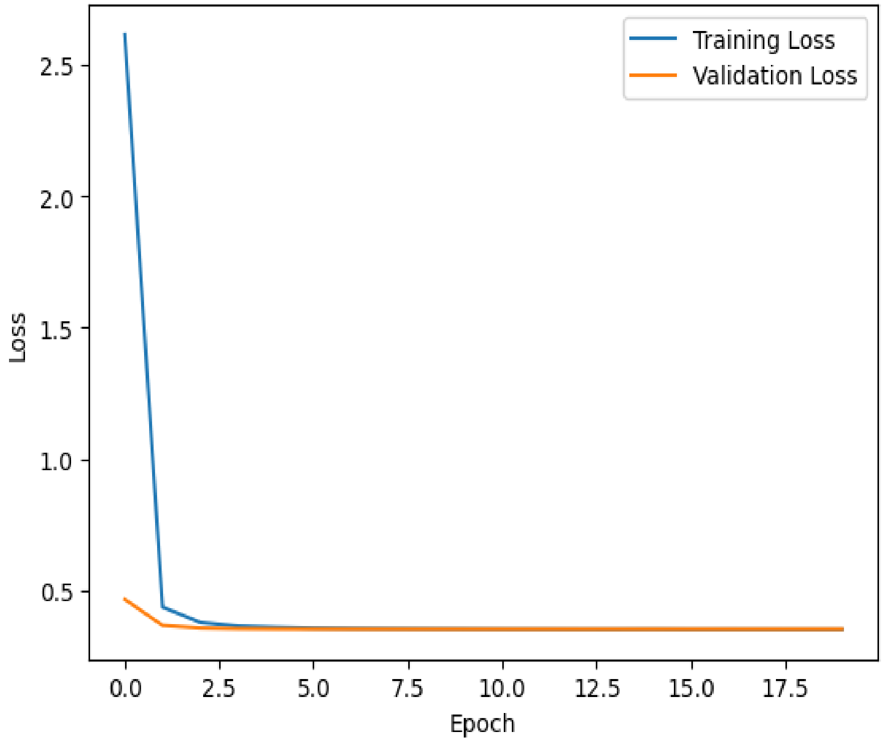


Figure 3.10: The proportions of the training modelTrain Test Splitl

and the finale result for this model lightFM for system Recommandation in this image 3.11 such as Training is Loss and Validation Loss is val_loss :

Loss=0.3520 , val_loss=0.3540

3.3.3 Model Conseption numbre -2- LightFM

In creating this model, I relied on a type of recommendation algorithm called LightFM

So wath is the LightFM ?

LightFM is a Python implementation of a number of popular recommendation algorithms for both implicit and explicit feedback. [23]

It also makes it possible to incorporate item and user metadata into traditional matrix factorization algorithms. It represents each user and item as the sum of the latent representations of their features, allowing recommendations to be generalized to new items (via item features) and to new users (via user features).

In LightFM, like in a collaborative filtering model, users and items are represented as latent vectors (embeddings). However, just as in a CB model, these are entirely defined by functions (in this case, linear combinations) of embeddings of the content features that describe each product or user. [20]

For example, if the movie 'Wizard of Oz' is described by the following features: 'musical fantasy', 'Judy Garland', and 'Wizard of Oz', then its latent representation will be given by the sum of these features' latent representations. In doing so, LightFM unites the advantages of contentbased and collaborative recommenders. [23]

3.3.3.1 The technologies used by LightFM

LightFM is a hybrid recommendation algorithm that combines collaborative filtering and content-based filtering techniques. Specifically, LightFM uses a form of collaborative filtering known as matrix factorization, along with content-based features, to generate recommendations. [24]

Below are details of the technologies used by LightFM:

- **Matrix Analysis :**

  LightFM uses matrix analysis to learn low-dimensional representations of users and items (or features). It decomposes the user-item interaction matrix into two lower-dimensional matrices: one for users and one for items. By learning these latent representations, LightFM captures underlying patterns in user-item interactions. [24]

- **Collaborative Filtering**

  In the context of LightFM, collaborative filtering refers to using user item interaction data to find similarities between users or items. By learning latent factors through matrix analysis, LightFM can identify similar users or items based on their interactions. This allows it to make recommendations to users based on users' preferences or similar items. [24]

- **Content-based filtering**

  LightFM also integrates content-based features into the recommendation process. These features describe additional characteristics of users and items beyond their interactions. For example, in a movie recommendation system, content-based features might include genre, director, or cast. By combining user item interaction data with content-based features, LightFM can provide personalized recommendations that take into account user preferences and item characteristics. [24]

- **Hybridization**

  One of LightFM's main strengths is its ability to seamlessly integrate collaborative filtering and content-based filtering techniques into a single model. By combining these approaches, LightFM can leverage the advantages of both methods and provide more accurate and diverse recommendations. [24]

In general, LightFM's recommendation technology involves learning latent representations of users and items through matrix analysis, incorporating collaborative filtering to capture user and item interactions, and incorporating content-based features to enhance the recommendation process.

3.3.3.2 Principle work of model lightFM

How lightFM is work ?

The LightFM paper describes beautifully how lightFM works. To put it simply in words, lightFM model learns embeddings (latent representations in a high-dimensional space) for users and items in a way that encodes user preferences over items. When multiplied together, these representations produce scores for every item for a given user; items scored highly are more likely to be interesting to the user.

The user and item representations are expressed in terms of representations of their features: an embedding is estimated for every feature, and these features are then summed together to arrive at representations for users and items . [20]

Description :

- The latent representation of user u is given by the sum of its features' latent vectors [20]:

$$q_u = \sum^{j \in f_u} \tag{3.1}$$

- And same for the items [20]:

$$q_i = \sum^{j \in f_i} \tag{3.2}$$

- The model's prediction for user u and item i is then given by the dot product of user and item representations, adjusted by user and item feature biases [20]:

$$r_{ui} = f(q_u \cdot p_i + b_u + b_i) \tag{3.3}$$

- The LightFM class represents a hybrid latent representation recommender model. [25]
- It learns embeddings for users and items, encoding user preferences over items. [25]
- The model expresses user and item representations in terms of their features, which are estimated for every feature and summed together to arrive at the representations. [25]
- Stochastic gradient descent methods are used to learn the embeddings. [25]
- Four loss functions are available: logistic, BPR (Bayesian Personalised Ranking), WARP (Weighted Approximate-Rank Pairwise), and k-OS WARP (k-th order statistic loss). [25]
- Two learning rate schedules are available: adagrad and adadelta. [25]

3.3.3.3 The purpose of using LightFM

Why LightFM ?

- In both cold-start and low-density scenarios, LightFM performs at least as well as pure content-based models, and significantly outperforms them when collaborative information is available in the training set or user features are included in the model. This is really useful for our recommendation system because we will have a lot of new tourists with unknown insights into places to visit which provides a very good environment for the cold start problem. [20]
- When collaborative data are abundant (warm start and dense user element matrix), LightFM performs highly efficiently. [20]
- The embeddings produced by LightFM encode important semantic information about features, and can be used for related recommendation tasks such as place recommendations. This is also very important for our problem. Because it is useful to find similar places so that this model can recommend predictions that contain places similar to professional places. [20]

3.3.3.4 LightFM Python Library

Fortunely, there is a library that makes easy to build a lightFM model. LightFM model is developed by Lyst. They also created a library for building lightfm model. It is very popular on Github having 2400+ stars and 226 closed issues. Because it is well maintained by Lyst(a london based e-commerce compnay) and it's learning community, lightFM python library is a really good source for building lightFM model. [20]

Benefit of LightFM python library: We can oviously make our implementation of lightFM model. But that will be reinvented the wheel. Because lightFM library is really well maintained library that are used production by many well reputed brand (Lyst, Sketchfab). [20]

The biggest benefit of lightfm library is that it implements**WARP (Weighted Approximate-Rank Pairwise) loss** for implicit feedback learning-to-rank. Wait! What is that?

For optimization of our matrix factorization function we can use different optimization methods e.g ALS, SGD. But there is another special optimization method called WARP (Weighted Approximate-Rank Pairwise). From the documentation, WARP works like these:

1. For a given (user, positive item pair), sample a negative item at random from all the remaining items. Compute predictions for both items; if the negative item's prediction exceeds that of the positive item plus a margin, perform a gradient update to rank the positive item higher and the negative item lower. If there is no rank violation, continue sampling negative items until a violation is found. [20]
2. If you found a violating negative example at the first try, make a large gradient update: this indicates that a lot of negative items are ranked higher than positives items given the current state of the model, and the model must be updated by a large amount. If it took a lot of sampling to find a violating example, perform a small update: the model is likely close to the optimum and should be updated at a low rate. [20]

It performs very well for implicit feedback model. Do you remember that our data is impicit feedback! For that reason, WARP loss is very essential for CareerVillage recommender system.

Becasue lightfm library implements this algorithm, that makes lightfm library really stands out.

Also, there are other important benefit also [20]:

1. LightFM is written in Cython and is paralellized via HOGWILD SGD. This will outperform any implementations of lightFM.
2. Already battle tested by many developers.
3. Already used in production by many well reputeted brand.
4. It's API is really developer friendly. This makes building model really easy.
5. Provide evaluation matrics for evaluating the perfomance of the model.
6. Finally, it is very very fast.

Building lightFM model from scracth by maintaing all of features above is really difficult and time consuming. For that reason, I am uisng LightFM python library for building the model.

How LightFM python library works: This library makes really easy for building lightFM model. There are couple of steps for building model using LightFM library. [20]

1. Process our Data and Make a lightFM dataset by using it's api
2. Build interaction matrix, user/item features
3. Make a model and train the model
4. Evaluate the model
5. Make predictions

3.3.3.5 Model Evaluation

The LightFM library provides a comprehensive set of evaluation functions to assess the performance of a fitted model. These functions are crucial for understanding how well the model generalizes to unseen data and for fine-tuning its parameters. Here are the key evaluation metrics available in the LightFM library [25]:

1. **ROC AUC Score :**

   This metric measures the area under the Receiver Operating Characteristic (ROC) curve, which represents the probability that a randomly chosen positive example has a higher score than a randomly chosen negative example. A perfect score is 1.0.

2. **Precision at K :**

   Precision at K measures the fraction of known positives in the first K positions of the ranked list of results. A perfect score is 1.0.

3. **Recall at K :**

   Recall at K measures the number of positive items in the first K positions of the ranked list of results divided by the number of positive items in the test period. A perfect score is 1.0.

4. **Reciprocal Rank :**

Reciprocal Rank measures 1 divided by the rank of the highest ranked positive example. A perfect score is 1.0.

These evaluation functions take into account various parameters such as the model, test interactions, train interactions (optional), user features, item features, and the number of threads for parallel computation. Additionally, they provide flexibility in preserving rows and checking intersections between test and train matrices to prevent optimistic ranks or wrong evaluation.

By leveraging these evaluation functions, users can gain valuable insights into the performance of their recommendation models and make informed decisions to improve them further.

3.3.3.6 Model training

The model training in LightFM involves optimizing the model parameters to minimize the chosen loss function while capturing meaningful relationships between users, items, and their features. By fine-tuning the training parameters and selecting appropriate loss functions, users can train models that perform well on their recommendation tasks.

The following figure 3.12 and figure 3.13 is an explanation of training the model in order to obtain a high accuracy value :

```
Epoch 1/100, AUC: 0.9994974732398987
Epoch 2/100, AUC: 0.9995141625404358
Epoch 3/100, AUC: 0.9995226860046387
Epoch 4/100, AUC: 0.9995245337486267
Epoch 5/100, AUC: 0.9995322823524475
Epoch 6/100, AUC: 0.9995377659797668
Epoch 7/100, AUC: 0.9995501637458801
Epoch 8/100, AUC: 0.9995524287223816
Epoch 9/100, AUC: 0.9995537400245667
Epoch 10/100, AUC: 0.999554455280304
Epoch 11/100, AUC: 0.9995622038841248
Epoch 12/100, AUC: 0.999582827091217
Epoch 13/100, AUC: 0.9995737671852112
Epoch 14/100, AUC: 0.9995859861373901
Epoch 15/100, AUC: 0.9995827078819275
Epoch 16/100, AUC: 0.9995989203453064
Epoch 17/100, AUC: 0.9995986819267273
Epoch 18/100, AUC: 0.9996081590652466
Epoch 19/100, AUC: 0.9996137619018555
Epoch 20/100, AUC: 0.9996194243431091
Epoch 21/100, AUC: 0.9996277093887329
Epoch 22/100, AUC: 0.9996402263641357
Epoch 23/100, AUC: 0.9996430277824402
Epoch 24/100, AUC: 0.9996508955955505
Epoch 25/100, AUC: 0.999648928642273
Epoch 26/100, AUC: 0.9996562600135803
Epoch 27/100, AUC: 0.9996671676635742
Epoch 28/100, AUC: 0.9996777176856995
Epoch 29/100, AUC: 0.9996902346611023
Epoch 30/100, AUC: 0.9996880888938904
```

```
Epoch 1/100, AUC: 0.9994974732398987
Epoch 2/100, AUC: 0.9995141625404358
Epoch 3/100, AUC: 0.9995226860046387
Epoch 4/100, AUC: 0.9995245337486267
Epoch 5/100, AUC: 0.9995322823524475
Epoch 6/100, AUC: 0.9995377659797668
Epoch 7/100, AUC: 0.9995501637458801
Epoch 8/100, AUC: 0.9995524287223816
Epoch 9/100, AUC: 0.9995537400245667
Epoch 10/100, AUC: 0.999554455280304
Epoch 11/100, AUC: 0.9995622038841248
Epoch 12/100, AUC: 0.999582827091217
Epoch 13/100, AUC: 0.9995737671852112
Epoch 14/100, AUC: 0.9995859861373901
Epoch 15/100, AUC: 0.9995827078819275
Epoch 16/100, AUC: 0.9995989203453064
Epoch 17/100, AUC: 0.9995986819267273
Epoch 18/100, AUC: 0.9996081590652466
Epoch 19/100, AUC: 0.9996137619018555
Epoch 20/100, AUC: 0.9996194243431091
Epoch 21/100, AUC: 0.9996277093887329
Epoch 22/100, AUC: 0.9996402263641357
Epoch 23/100, AUC: 0.9996430277824402
Epoch 24/100, AUC: 0.9996508955955505
Epoch 25/100, AUC: 0.999648928642273
Epoch 26/100, AUC: 0.9996562600135803
Epoch 27/100, AUC: 0.9996671676635742
Epoch 28/100, AUC: 0.9996777176856995
Epoch 29/100, AUC: 0.9996902346611023
Epoch 30/100, AUC: 0.9996880888938904
```

```
Epoch 61/100, AUC: 0.9998026490211487
Epoch 62/100, AUC: 0.9997969269752502
Epoch 63/100, AUC: 0.9998020529747009
Epoch 64/100, AUC: 0.9998071193695068
Epoch 65/100, AUC: 0.9998183250427246
Epoch 66/100, AUC: 0.9998108744621277
Epoch 67/100, AUC: 0.9998102784156799
Epoch 68/100, AUC: 0.9998258352279663
Epoch 69/100, AUC: 0.9998254179954529
Epoch 70/100, AUC: 0.9998264312744141
Epoch 71/100, AUC: 0.999830424785614
Epoch 72/100, AUC: 0.9998329877853394
Epoch 73/100, AUC: 0.999840497970581
Epoch 74/100, AUC: 0.9998442530632019
Epoch 75/100, AUC: 0.9998534321784973
Epoch 76/100, AUC: 0.9998531341552734
Epoch 77/100, AUC: 0.99985271692276
Epoch 78/100, AUC: 0.9998545050621033
Epoch 79/100, AUC: 0.9998623728752136
Epoch 80/100, AUC: 0.9998646974563599
Epoch 81/100, AUC: 0.9998633861541748
Epoch 82/100, AUC: 0.999871015548706
Epoch 83/100, AUC: 0.999873161315918
Epoch 84/100, AUC: 0.9998746514320374
Epoch 85/100, AUC: 0.9998711943626404
Epoch 86/100, AUC: 0.9998789429664612
Epoch 87/100, AUC: 0.9998815655708313
Epoch 88/100, AUC: 0.9998890161514282
Epoch 89/100, AUC: 0.9998888969421387
Epoch 90/100, AUC: 0.999903678894043
```

```
Epoch 91/100, AUC: 0.9999093413352966
Epoch 92/100, AUC: 0.9999086260795593
Epoch 93/100, AUC: 0.9999070167541504
Epoch 94/100, AUC: 0.9999035596847534
Epoch 95/100, AUC: 0.9999060034751892
Epoch 96/100, AUC: 0.9999155402183533
Epoch 97/100, AUC: 0.9999145269393921
Epoch 98/100, AUC: 0.999908447265625
Epoch 99/100, AUC: 0.9999194145202637
Epoch 100/100, AUC: 0.9999179840087891
```

Figure 3.12: Training the model

In the following image 3.14 is a data chart representing the proportions of the training model:

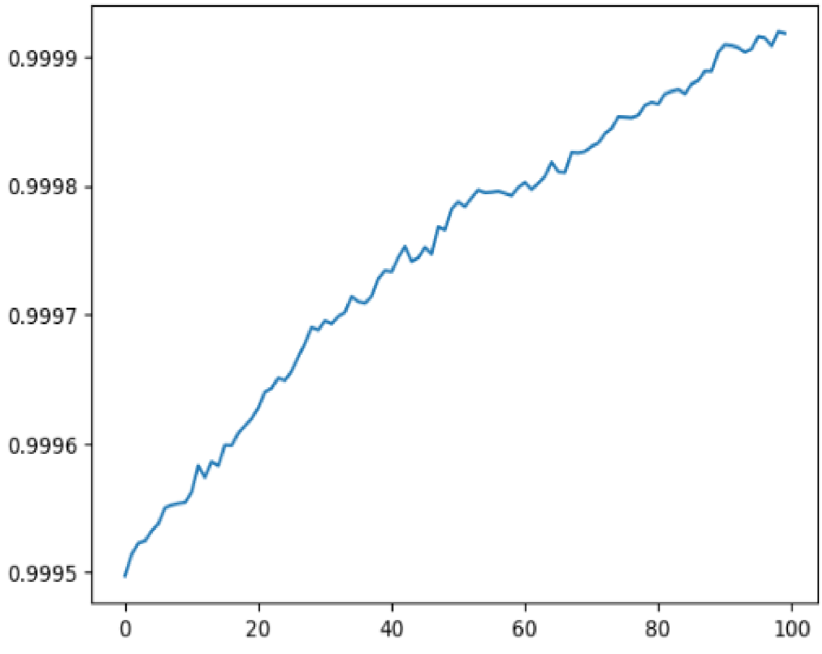


Figure 3.13: The proportions of the training modelTrain Test Split

And the finale result for this model lightFM for system Recommandation in this image 3.15

```
AUC Score: 0.999918
```

Figure 3.14: The proportions of the training modelTrain Test Split

Finally, by reaching a high accuracy rate, we can say that the model gives correct results with high accuracy

3.3.4 Comparison between models :

Here we show the recommendation result for user 20 for both models 1 and 2 using different techniques in the following table :

	Model-1-(RecommenderNet)	**Model-2-LightFM**
Result	sindu kusuma Edupark (SKE) jurang tembelan kanigoro taman balai kota bandung jembatan pasupati pasar baru taman hutan raya ir . h. juanda bandors city tour Museum de javasche bank Mounuemen tugu pahlawan Moumen bambu rancing surabaya	Kota Tua Museum Bank Indonesia Wisata Kuliner Pecenongan Grand Indonesia Mall Nol Kilometer Jl.Malioboro Alun-Alun Kota Bandung Lawangwangi Creative Space Gunung Lalakon

Table 3.1: Comparison between Result models for user 20

3.4 Diagrams of Model Training

Unified Modeling Language (UML) is a unified modeling language used in software engineering to visually represent software systems. Provides a set of graphical symbols to describe a system's structure, behavior, and interactions. UML diagrams serve as a common language for communication between software developers, designers, stakeholders, and other project members throughout the software development life cycle.

In this section, we are going to provide a presentation of our project with a diagram use case and diagram sequence, also diagram of class for more understanding:

3.4.1 use case diagrame

Describe the interactions between actors (users or external systems) and the system to achieve specific goals. In the following figure 3.16 we explain the above

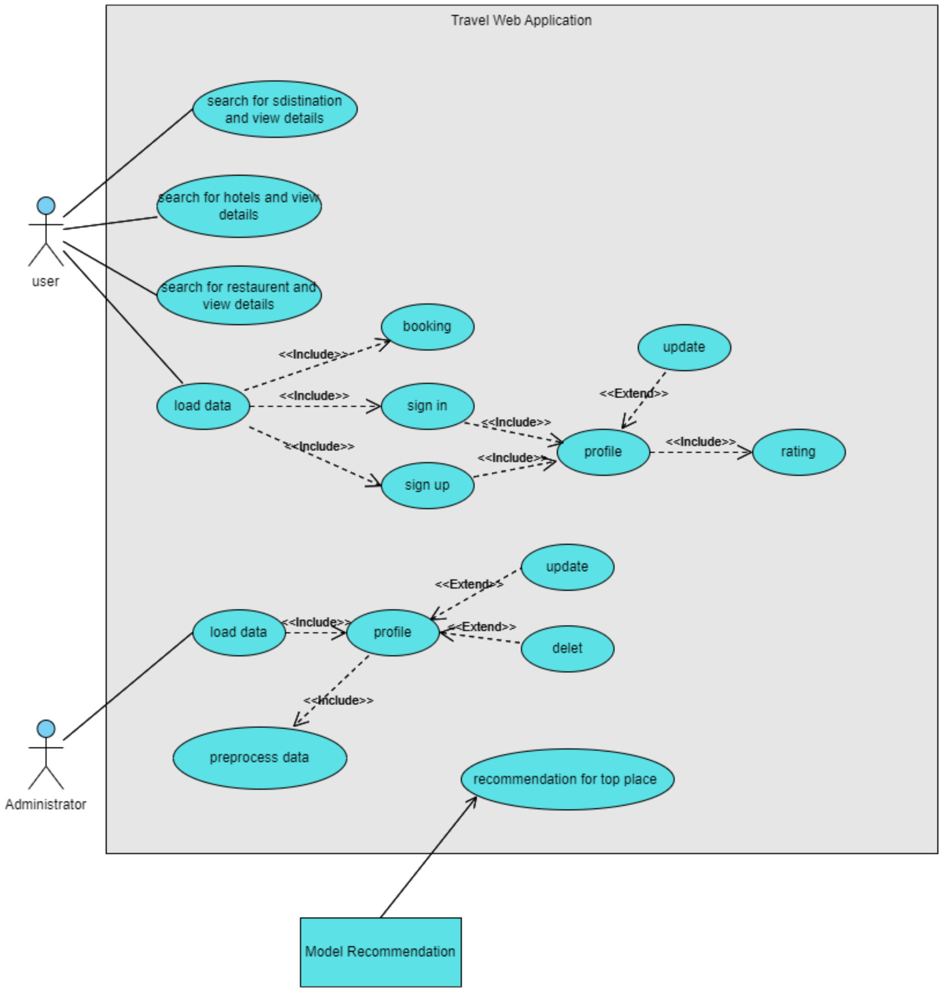


Figure 3.15: Use Case Diagram of The System.

3.4.2 Sequence Diagram

3.4.2.1 sequence diagram (sign in or sign up)

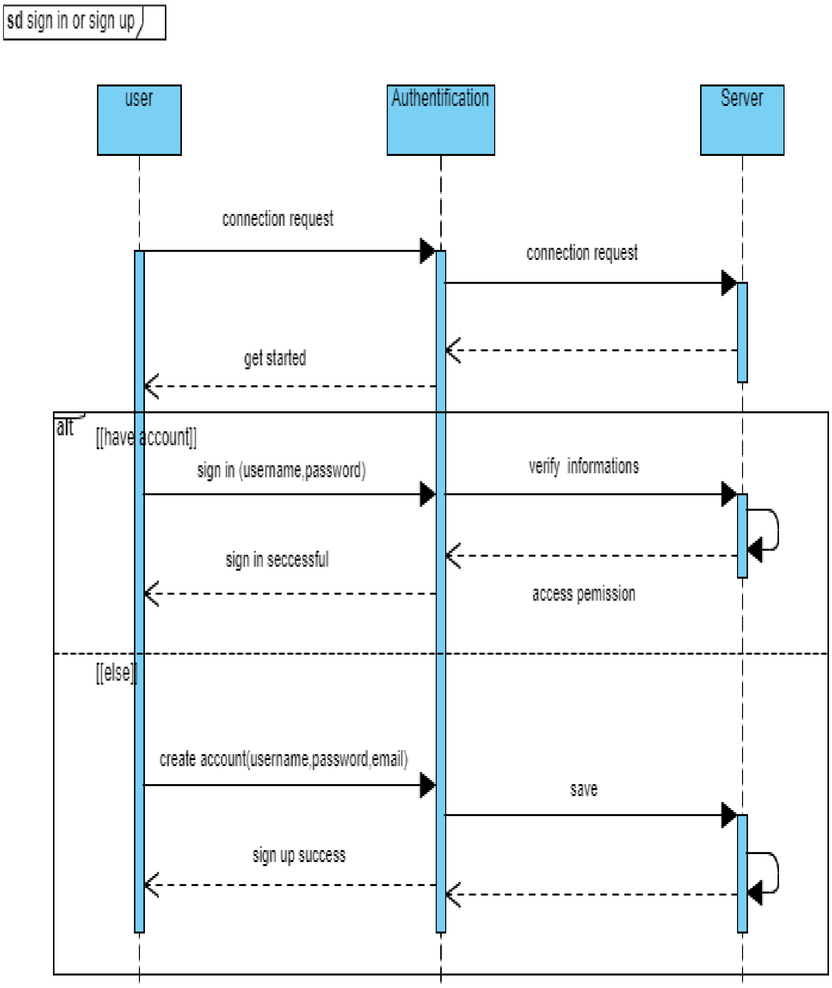


Figure 3.16: Sequence Diagram of sign in or sign up

3.4.2.2 sequence diagram (administrator sitting)

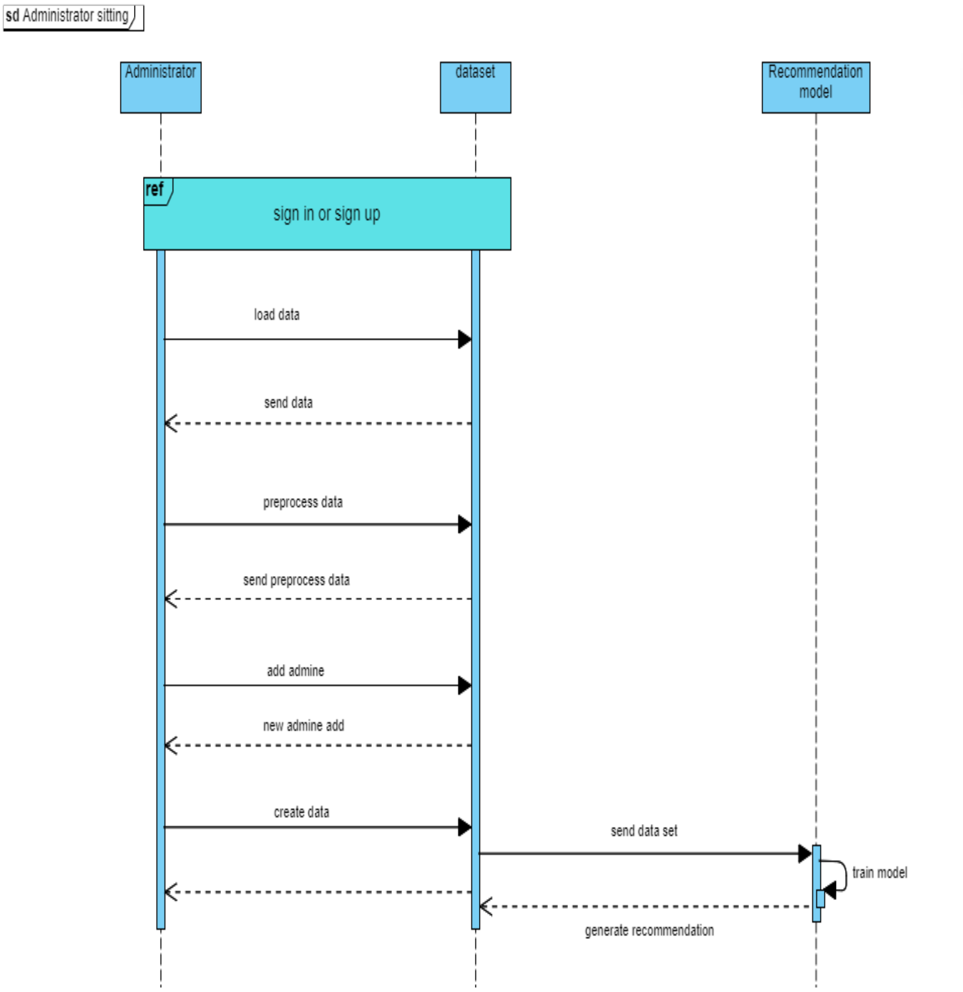


Figure 3.17: Sequence Diagram of Administratoe sitting

3.4.2.3 Sequence diagram for travel web application

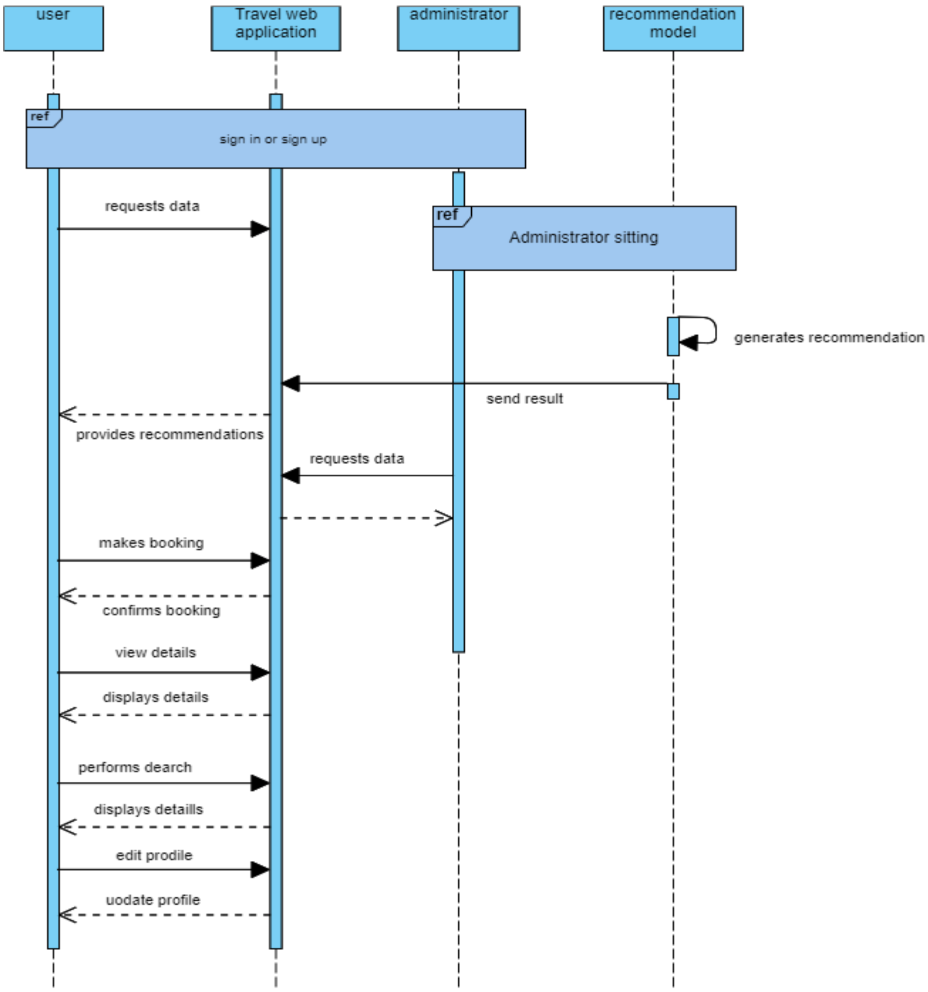


Figure 3.18: Sequence Diagram

3.4.3 diagram of class

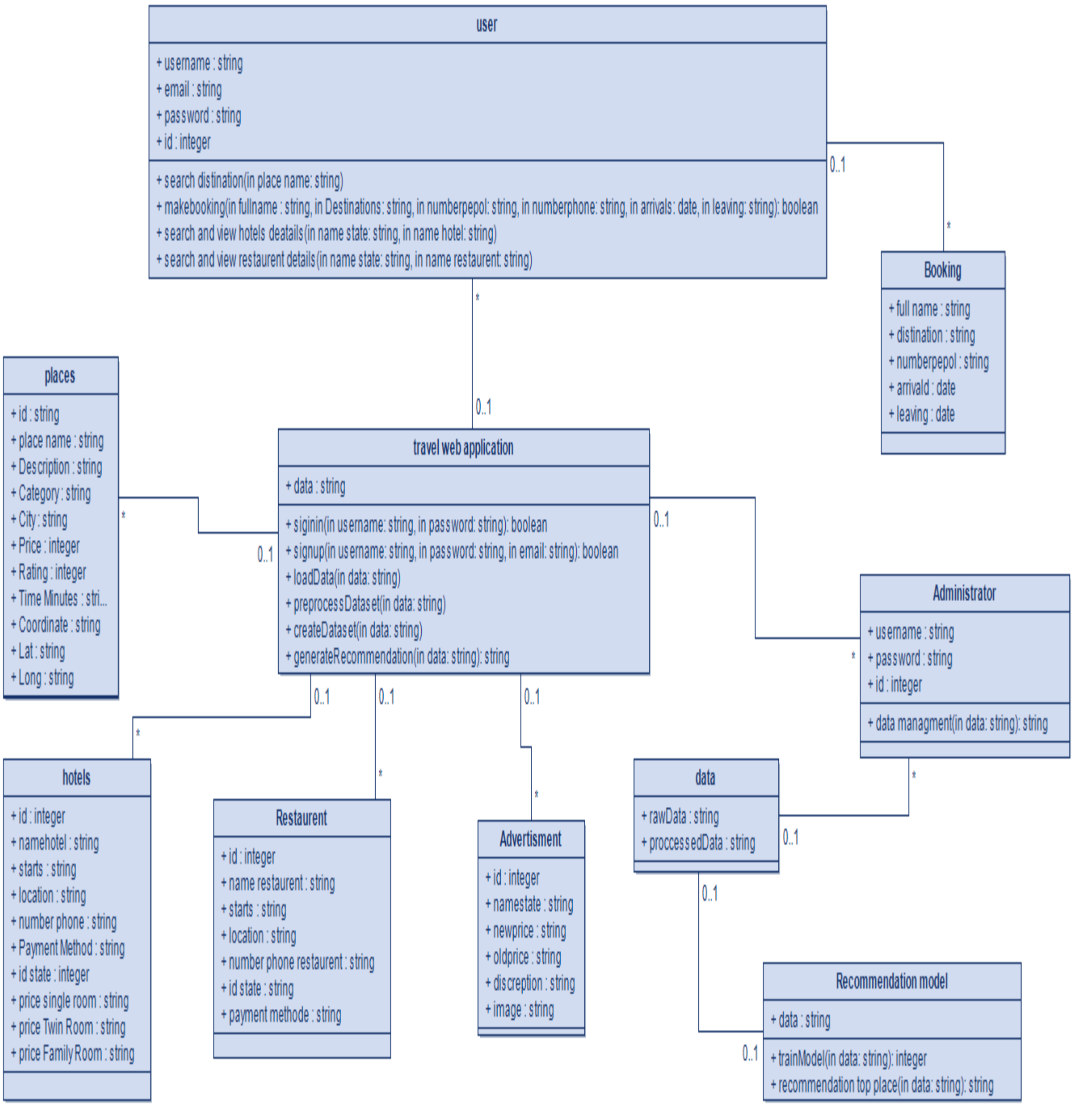


Figure 3.19: Sequence Diagram

3.5 conclution

In this chapter we have presented and described our solution in detail. First we have presented our general architecture and then we have moved on to our detailed architecture which shows the actors in our system, we have introduced each component and how it works and we have added some algorithms and flowcharts used. After that, we have presented its functionality in the form of mock algorithms, then in the form of use cases and sequence diagrams. In the next chapter we will represent the implementation of our system.

4

Implementation

4.1 Introduction

In this chapter, our primary focus will be on the implementation and practical aspects of our project we have developed. We will begin by introducing the tools we have utilized, including software, hardware, and applications.

4.1.1 HTML

HTML is an acronym for HyperText Markup Language. HTML documents, the foundation of all content appearing on the World Wide Web (WWW), consist of two essential parts: information content and a set of instructions that tells a computer how to display that content. The instructions, the "markup," in editorial jargon, comprise the HTML language. It is not a programming language in the traditional sense but rather a set of instructions about how to display content. The computer application that translates this description is called a Web browser. Ideally, online content should always look the same regardless of the browser used or the operating system on which it resides, but the goal of platform independence is achieved only approximately in practice. [26]

Figure 4.1: HTML

4.1.2 CSS

CSS stands for Calderbank-Shor-Steane codes, which are a type of quantum error-correcting code used in quantum computing [27]

- They are a subclass of stabilizer codes, which are a class of quantum codes that can be defined using a set of generators of the Pauli group on n qubits
- CSS codes are particularly useful for protecting quantum information against decoherence, which is the loss of quantum coherence due to interactions with the environment In the context of quantum computing, CSS codes play a crucial role in the construction of magic distillation protocols, which are essential for fault-tolerant quantum computing
- They have been used to obtain distillation bounds that can outperform previous monotone bounds in regimes of practical interest

Figure 4.2: CSS

4.1.3 PHP

PHP is a popular general-purpose scripOng language that is especially suited to web development. —php.net Recursive acronym: ”PHP: PHP Hypertext Preprocessor” w3techs.com reports that PHP is used on 81.2% of all websites php.net claims PHP ”is installed on” 244 million websites PHP is used by Facebook, Wikipedia, Twiter and many other large sites. PHP underlies many Content Management Systems including Drupal and WordPress. [28]

Figure 4.3: PHP

4.1.4 XAMPP

XAMPP is meant only for development purposes. It has certain configuration settings that make it easy to develop locally but that are insecure if you want to have your installation accessible to others. If you want have your XAMPP accessible from the internet, make sure you understand the implications and you checked the FAQs to learn how to protect your site. Alternatively you can use WAMP, MAMP or LAMP which are similar packages which are more suitable for production. [29]

Figure 4.4: XAMPP

4.1.5 Python

Is an object-oriented, interpreted, mid-level programming language that is simple to learn and use, and it is today regarded as one of the best programming languages to learn. Some of the reasons for its success include its free, open-source nature and large online community. [30]

Figure 4.5: Python

4.1.6 Java Script

JavaScript é a linguagem de programação da Web. JavaScript faz parte da tríade de tecnologias que todos os desenvolvedores Web devem conhecer: HTML, para especificar o conteúdo de páginas Web; CSS, para especificar a apresentação dessas páginas; e JavaScript, para especificar o comportamento delas
Se você já conhece outras linguagens de programação, talvez ajude saber que JavaScript é uma linguagem de alto nível, dinâmica, interpretada e não tipada, conveniente para estilos de programação orientados a objetos e funcionais. [31]

Figure 4.6: JavaScript

4.1.7 Colaboratory

Colaboratory is a free Jupyter notebook environment provided by Google where you can use free GPUs and TPUs which can solve all these issues. It contains almost all the modules you need for data science analysis. These tools include but are not limited to Numpy, Scipy, Pandas, etc. Even deep learning frameworks, such as Tensorflow, Keras and Pytorch are also included. [32]

Figure 4.7: Colaboratory

4.1.8 Kaggle

Kaggle allows users to find datasets they want to use in building AI models, publish datasets, work with other data scientists and machine learning engineers, and enter competitions to solve data science challenges.
Kaggle got its start in 2010 by offering machine learning and data science competitions as well as offering a public data and cloud-based business platform for data science and AI education.. [33]

Figure 4.8: Kaggle

4.2 Machine learning Model Implementation

This part explains the machine learning process to recommend the best places. To do this, we used kaggle platform that allows the use of Notebook to execute forms.

First we must install a library LightFM :

```
pip install lightfm
```

Figure 4.9: install LightFM -1-

```
Collecting lightfm
  Downloading lightfm-1.17.tar.gz (316 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 316.4/316.4 kB 3.7 MB/s eta 0:00:00a 0:00:01
  Preparing metadata (setup.py) ... done
Requirement already satisfied: numpy in /opt/conda/lib/python3.10/site-packages (from lightfm) (1.26.4)
Requirement already satisfied: scipy>=0.17.0 in /opt/conda/lib/python3.10/site-packages (from lightfm) (1.11.
4)
Requirement already satisfied: requests in /opt/conda/lib/python3.10/site-packages (from lightfm) (2.31.0)
Requirement already satisfied: scikit-learn in /opt/conda/lib/python3.10/site-packages (from lightfm) (1.2.2)
Requirement already satisfied: charset-normalizer<4,>=2 in /opt/conda/lib/python3.10/site-packages (from requ
ests->lightfm) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /opt/conda/lib/python3.10/site-packages (from requests->lightf
m) (3.6)
Requirement already satisfied: urllib3<3,>=1.21.1 in /opt/conda/lib/python3.10/site-packages (from requests->
lightfm) (1.26.18)
Requirement already satisfied: certifi>=2017.4.17 in /opt/conda/lib/python3.10/site-packages (from requests->
lightfm) (2024.2.2)
Requirement already satisfied: joblib>=1.1.1 in /opt/conda/lib/python3.10/site-packages (from scikit-learn->l
ightfm) (1.4.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in /opt/conda/lib/python3.10/site-packages (from scikit-l
earn->lightfm) (3.2.0)
Building wheels for collected packages: lightfm
  Building wheel for lightfm (setup.py) ... done
  Created wheel for lightfm: filename=lightfm-1.17-cp310-cp310-linux_x86_64.whl size=464713 sha256=77635d1984
ab63443efe03d70ce5afc081949c922ff591e7a8faad8122985b99
  Stored in directory: /root/.cache/pip/wheels/4f/9b/7e/0b256f2168511d8fa4dae4fae0200fdbd729eb424a912ad636
Successfully built lightfm
Installing collected packages: lightfm
Successfully installed lightfm-1.17
Note: you may need to restart the kernel to use updated packages.
```

Figure 4.10: install LightFM -2-

next, we imported what was needed The beams shown in Figure 4.11

```python
# Importing pandas library and aliasing it as pd
import pandas as pd

# Importing numpy library and aliasing it as np
import numpy as np

# Importing Path class from pathlib module
from pathlib import Path

# Importing pyplot module from matplotlib library and aliasing it as plt
import matplotlib.pyplot as plt

# Importing LightFM class from lightfm library
from lightfm import LightFM

# Importing Dataset class from lightfm.data module
from lightfm.data import Dataset

# Importing auc_score function from lightfm.evaluation module
from lightfm.evaluation import auc_score
```

Figure 4.11: Import packages

4.2.1 Dataset Import and Preparation

We collected the data set "Tourism Destination" and modified it according to what we needed. The dataset contains 3 files :

tourism rating:

tourism_rating.csv contains 3 columns, namely the user, the place, and the rating given, serves to create a recommendation system based on the rating ,As shown in the following image 4.12

	User_Id	Place_Id	Place_Ratings
0	1	179	3
1	1	344	2
2	1	5	5
3	1	373	3
4	1	101	4
...	...	...	...
9995	300	425	2
9996	300	64	4
9997	300	311	3
9998	300	279	4
9999	300	163	2

10000 rows × 3 columns

Figure 4.12: Our Dataset -1-

Tourism_with_id:

This dataset contains data for 400 tourist attractions from 5 large cities in Indonesia, As shown in the following image 4.13

	User_Id	Place_Id	Place_Ratings	Place_Name	Description	City	Category
0	1	179	3	Candi Ratu Boko	Situs Ratu Baka atau Candi Boko (Hanacaraka:[illegible]...	Yogyakarta	Budaya
1	1	344	2	Pantai Marina	Pantai Marina (bahasa Jawa: [illegible], trans...	Semarang	Bahari
2	1	5	5	Atlantis Water Adventure	Atlantis Water Adventure atau dikenal dengan A...	Jakarta	Taman Hiburan
3	1	373	3	Museum Kereta Ambarawa	Museum Kereta Api Ambarawa (bahasa Inggris: In...	Semarang	Budaya
4	1	101	4	Kampung Wisata Sosro Menduran	Kampung wisata Sosromenduran merupakan kampung...	Yogyakarta	Budaya
...	...	...	...	...	...	...	...

9995	300	425	2	Waterpark Kenjeran Surabaya	Waterpark Kenjeran Surabaya merupakan wisata k...	Surabaya	Taman Hiburan
9996	300	64	4	Museum Sasmita Loka Ahmad Yani	Museum Sasmita Loka Ahmad Yani adalah salah sa...	Jakarta	Budaya
9997	300	311	3	The Lodge Maribaya	The Lodge Maribaya adalah salah satu tempat wi...	Bandung	Cagar Alam
9998	300	279	4	Masjid Agung Trans Studio Bandung	Masjid Agung Trans Studio Bandung (TSB) berdir...	Bandung	Tempat Ibadah
9999	300	163	2	Watu Mabur Mangunan	Kawasan Tebing Watu Mabur ini terbilang belum ...	Yogyakarta	Cagar Alam

Figure 4.13: Our Dataset -2-

user:

user.csv which contains dummy user data to generate recommendation features based on the user, As shown in the following image 4.14

User_Id	Location	Age	username	password	email
1	Semarang, .	20	use1	1	use1@gmail.com
2	Bekasi, Jaw	21	use2	2	use2@gmail.com
3	Cirebon, Ja	23	use3	3	use3@gmail.com
4	Bekasi, Jaw	21	use4	4	use4@gmail.com
5	Lampung, S	20	use5	5	use5@gmail.com
6	Jakarta Uta	18	use6	6	use6@gmail.com
7	Jakarta Sela	39	use7	7	use7@gmail.com
8	Bandung, Ja	40	use8	8	use8@gmail.com
9	Surabaya, J	38	use9	9	use9@gmail.com
10	Bekasi, Jaw	39	use10	10	use10@gmail.com
11	Yogyakarta,	20	use11	11	use11@gmail.com
12	Bogor, Jawa	37	use12	12	use12@gmail.com
13	Depok, Jaw	18	use13	13	use13@gmail.com
14	Jakarta Pus	26	use14	14	use14@gmail.com

....	....	....	....	....	......
300	Ponorogo,.	26	use300	300	use300@gmail.com

Figure 4.14: Our Dataset -3-

To use this dataset, the first step is to import it as shown in Figure 4.15

```
# Define constants
DATA_PATH = "/kaggle/input/indonesia-tourism-destination"

# Load data
def load_data():
  info_tourism = pd.read_csv(f"{DATA_PATH}/tourism_with_id.csv")
  tourism_rating = pd.read_csv(f"{DATA_PATH}/tourism_rating.csv")
  users = pd.read_csv(f"{DATA_PATH}/user.csv")
  return info_tourism, tourism_rating, users
```

Figure 4.15: Import Our Dataset

Then to prepare the dataset we split it into Features/Target as shown in Figure 4.16 Function to split data into training and validation sets

After, we resized the distribution of features values so that the mean of the observed values is 0 and the standard deviation is 1 with StandardScaler and split the dataset into training sets 80% and testing sets 20% which we chose as the best splitting, with convert the data . as shown in Figure 4.17

```
user_items = df[['user', 'place']].values
ratings = df['Place_Ratings'].values.astype(np.float32)
```

Figure 4.16: Dataset Split Code

```
  # Split user-place interactions and ratings into training and validation sets
train_user_items, val_user_items, train_ratings, val_ratings = (
    user_items[:train_indices],
    user_items[train_indices:],
    ratings[:train_indices],
    ratings[train_indices:]
)
```

Figure 4.17: Dataset Preparation

- train_indices calculates the index to split the data into 80% training and 20% validation.
- train_user_items contains the first 80% of user_items for training.
- val_user_items contains the remaining 20% of user_items for validation.
- train_ratings contains the first 80% of ratings for training. val_ratings contains the remaining 20% of ratings for validation.

To explain more about how to predict A matrix is a representation of interactions between users and items, where each row represents a user and each column represents an item. However, the values in this matrix are not only binary (0 and 1), but seem to indicate the strength or intensity of interaction between users and items,as shown in Figure 4.18

```
Adjacency Matrix:
[[1 0 0 ... 0 0 0]
 [0 1 0 ... 0 0 0]
 [0 1 2 ... 0 0 0]
 ...
 [0 0 1 ... 0 0 1]
 [0 0 0 ... 2 0 0]
 [0 0 0 ... 0 0 0]]
```

Figure 4.18: interactions matrix

- Rows correspond to users, and columns correspond to items.
- The value at position (i, j) indicates the strength or intensity of the interaction between user i and item j.
- A value of 0 indicates no interaction between the user and the item.
- Non-zero values indicate a certain level of interaction, with higher values typically representing stronger interactions.

For example:

- Row 1 (User 1) has strong interaction (value 2) with item 3 (column 3), but no interaction with other items.
- Row 2 (User 2) has strong interaction (value 2) with item 2 (column 2), but no interaction with other items.
- Row 3 (User 3) has moderate interaction (value 2) with item 2 (column 2) and strong interaction (value 3) with item 3 (column 3), but no interaction with other items.
- Row n (user n) contains interactions of different strengths with different items.

This is a simple exemple when the data is like thise , as shown in Figure 4.19 and 4.20 :

```
data = {
    'User_Id': [1, 2, 3, 4, 5],
    'Place_Id': [101, 102, 103, 104, 105],
    'Place_Ratings': [5, 3, 4, 2, 1]
}
df = pd.DataFrame(data)
```

Figure 4.19: Create a sample dataframe

```
Adjacency Matrix:
[[1 0 0 0 0]
 [0 1 0 0 0]
 [0 0 1 0 0]
 [0 0 0 1 0]
 [0 0 0 0 1]]
```

Figure 4.20: interactions matrix of the Previous example

The adjacency matrix you've printed represents the interactions between users and items, where each row corresponds to a user and each column corresponds to an item. Here's how to interpret the matrix:

- Rows represent users, and columns represent items.

- The value of 1 at position (i, j) indicates that user i interacted with item j.

- The value of 0 at position (i, j) indicates that user i did not interact with item j.

Interpreting the given adjacency matrix:

- Row 1 (user 1) interacts with item 1 (column 1).
- Row 2 (user 2) interacts with item 2 (column 2).
- Row 3 (user 3) interacts with item 3 (column 3).
- Row 4 (user 4) interacts with item 4 (column 4).
- Row 5 (user 5) interacts with item 5 (column 5).

All other entries in the matrix are 0, indicating no interaction between those users and items.

Where 1 indicates engagement (for example, purchase, click, rating) and 0 indicates no engagement.

4.2.2 Training the ML Model

Machine learning algorithms were used lightFM

```
model.fit(interactions, epochs=100,verbose=True)
```

```
Epoch: 100%|██████████| 100/100 [00:01<00:00, 69.05it/s]
<lightfm.lightfm.LightFM at 0x7fcd51ce1f60>
```

Figure 4.21: Define and train the LightFM model

```
epochs = 100
adadelta_auc = []

for epoch in range(epochs):
    model.fit_partial(interactions, epochs=1)
    adadelta_auc.append(auc_score(model, interactions).mean())
```

Figure 4.22: Training the ML Model

In the end we have this result for acc:

```
AUC Score: 0.9995267
```

Figure 4.23: Accuracy ratio

4.2.3 The API server

4.2.3.1 difintion

An API server is a type of software that allows different applications to communicate with one another. It is an intermediary between different systems, allowing them to share data and functionality. This can include providing access to data, automating a process, or acting as a backend for an application.

The lightFM model will be saved to a JSON file with the pickle pip installed. To connect it with the web application, as shown in Figure 4.24

```python
# Load model
with open('saved_model.pkl', 'rb') as f:
    model = pickle.load(f)
```

Figure 4.24: Saving Code of LightFM Model

- The recommender system model works with API and JSON files, using Python and Flask
- We created an API for the recommendation service using the trained model
- Which performs JSON processing: reading and writing data in JSON format for input and output

4.3 Web Application Interface

4.3.1 front face

4.3.1.1 Logine Page :

Every user must have an account to access the application. Anyone who has an account can log in with the user name and password as shown in the image 4.25

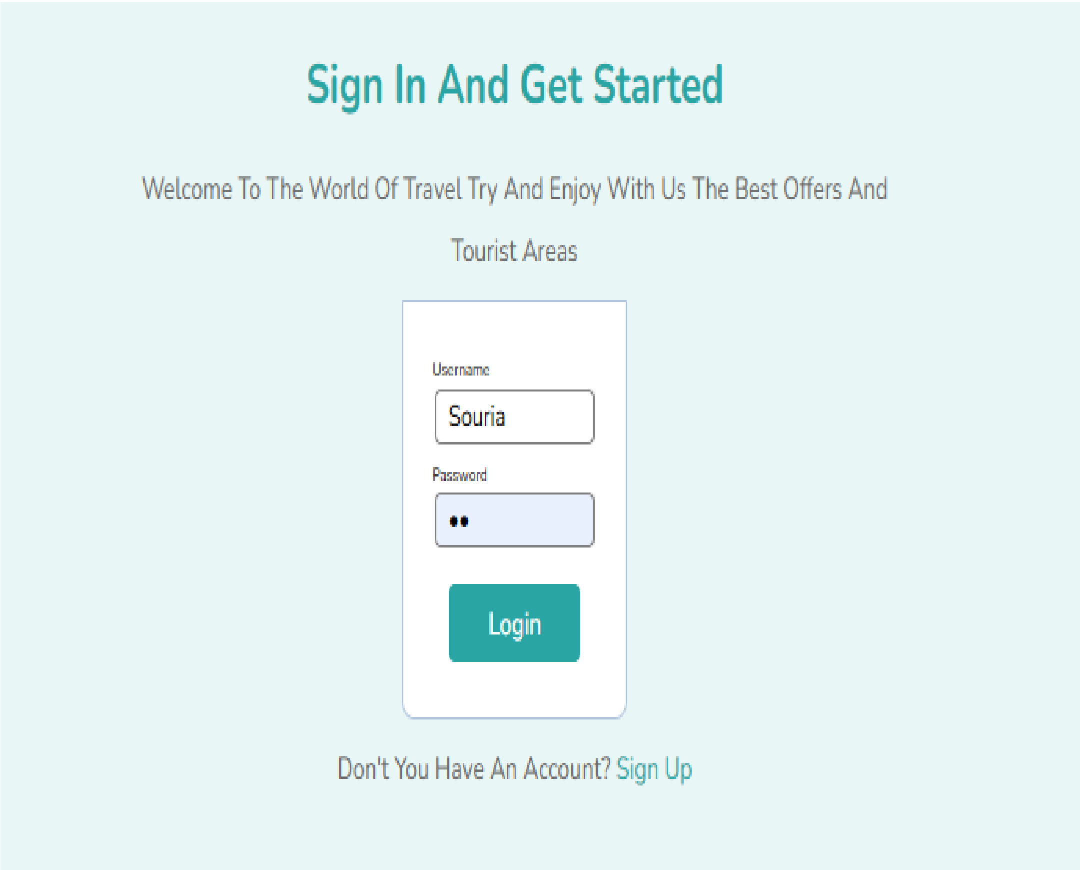


Figure 4.25: Login Page

4.3.1.2 register page :

For people who do not have an account, they can create an account with us in simple steps, as shown in the image 4.26

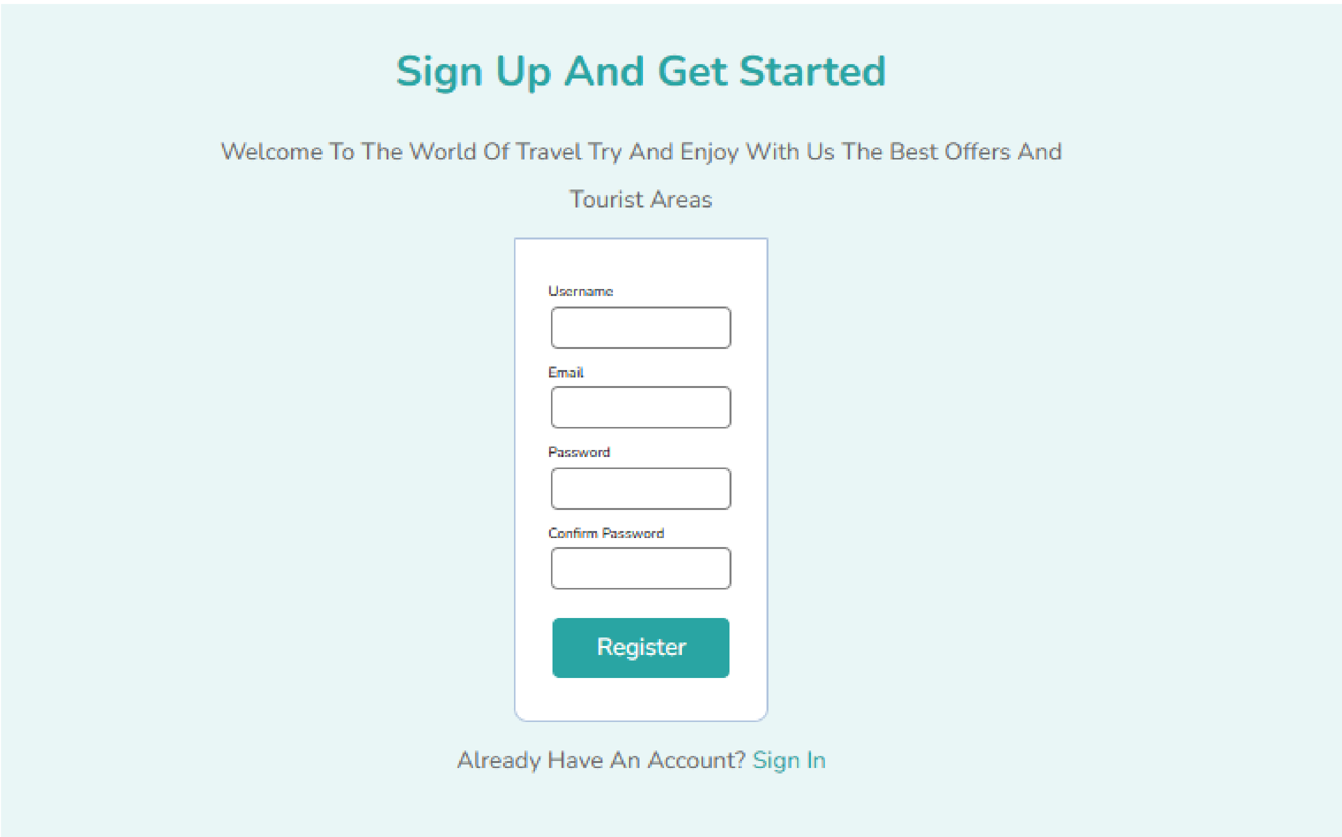


Figure 4.26: register Page

When entering the platform, there is a part for the user and another for the administrator, explained as follows :

4.3.2 Admin

4.3.2.1 Dashboard :

The interface will contain collections of the total number of all users, the total number of administrators, the total number of hotels in the application, the total number of existing restaurants, the total number of places, the total number of reservations made by each user, and the total number of advertisements, as shown in the following image 4.27 :

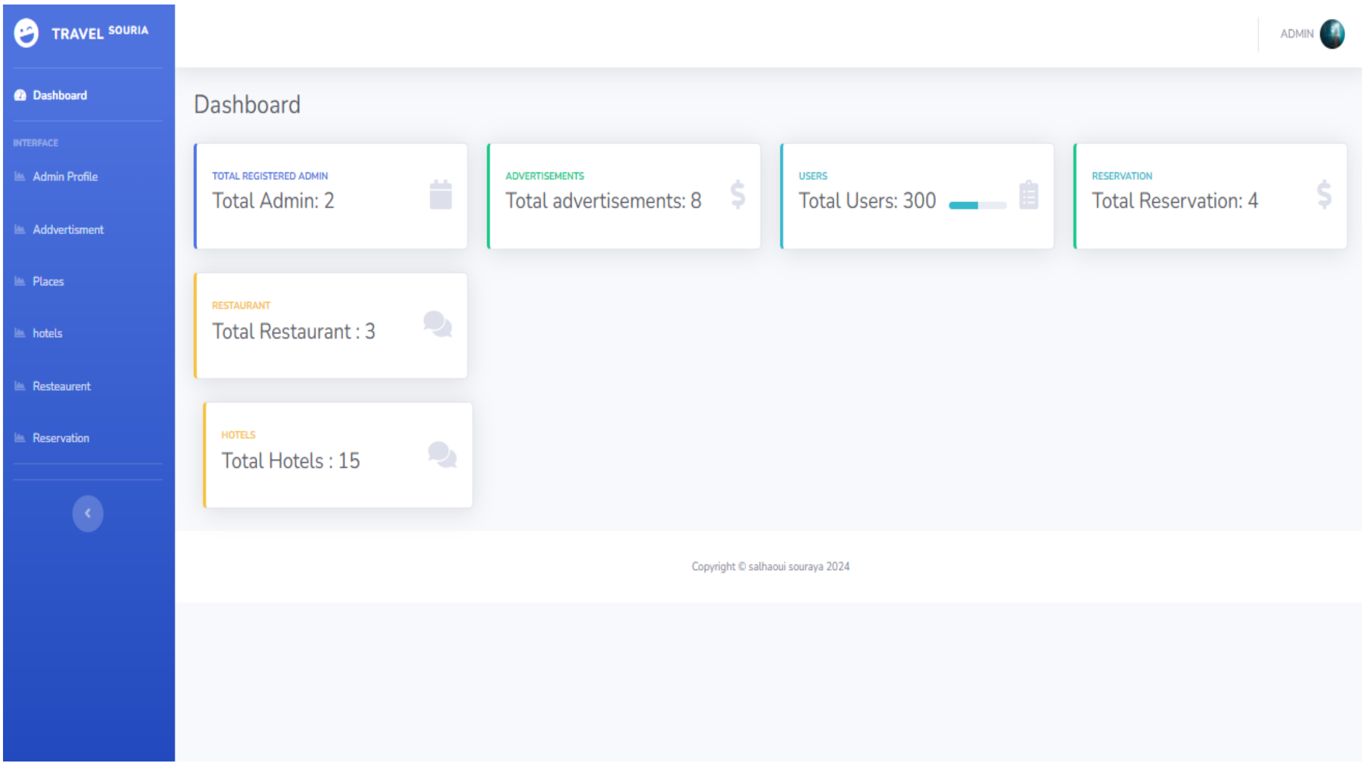


Figure 4.27: Dashboard Page

4.3.2.2 Admin Profile :

In this part of the platform, we find a table containing all the administrators who can modify the application data with their private information, such as a user name, email, and password, as shown in image 4.28, so that new officials can be added, shown in image 4.29, by adding their information represented by a user name and Email, password, and confirm the password.

Also, each administrator can modify his personal file by clicking on the edit button, and the page for modification is shown in image 4.30,and there is also the feature to delete an administrator.

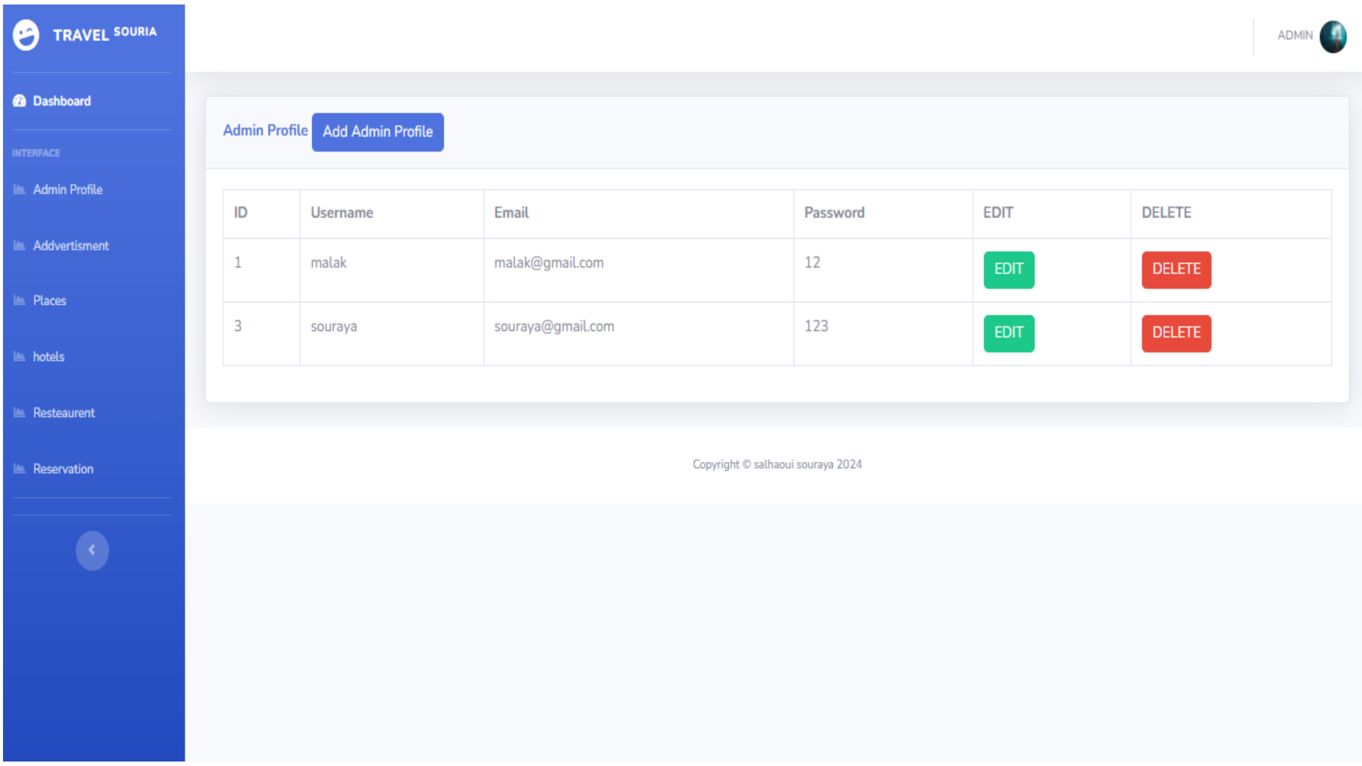


Figure 4.28: Admin Profile

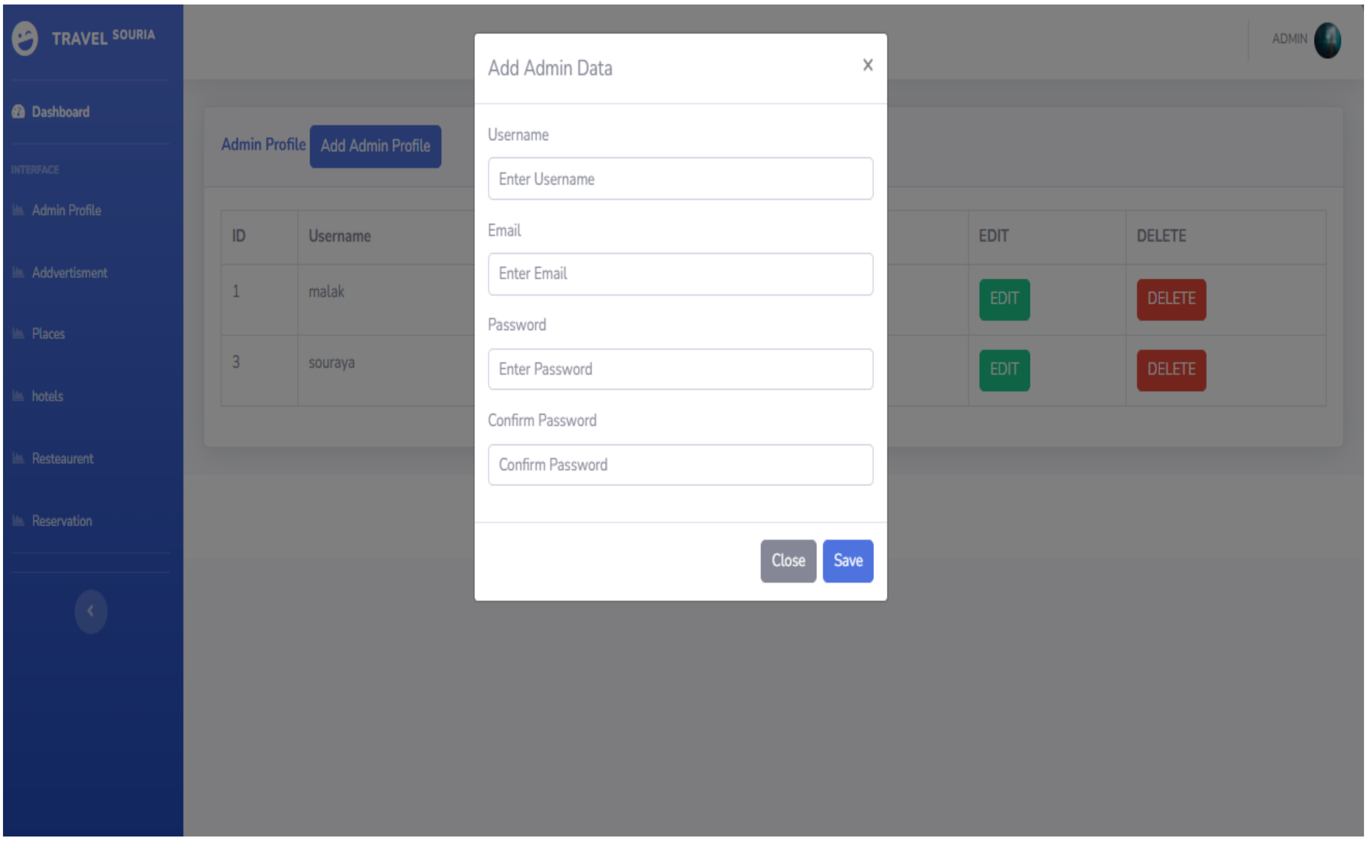


Figure 4.29: Add Admin Profile

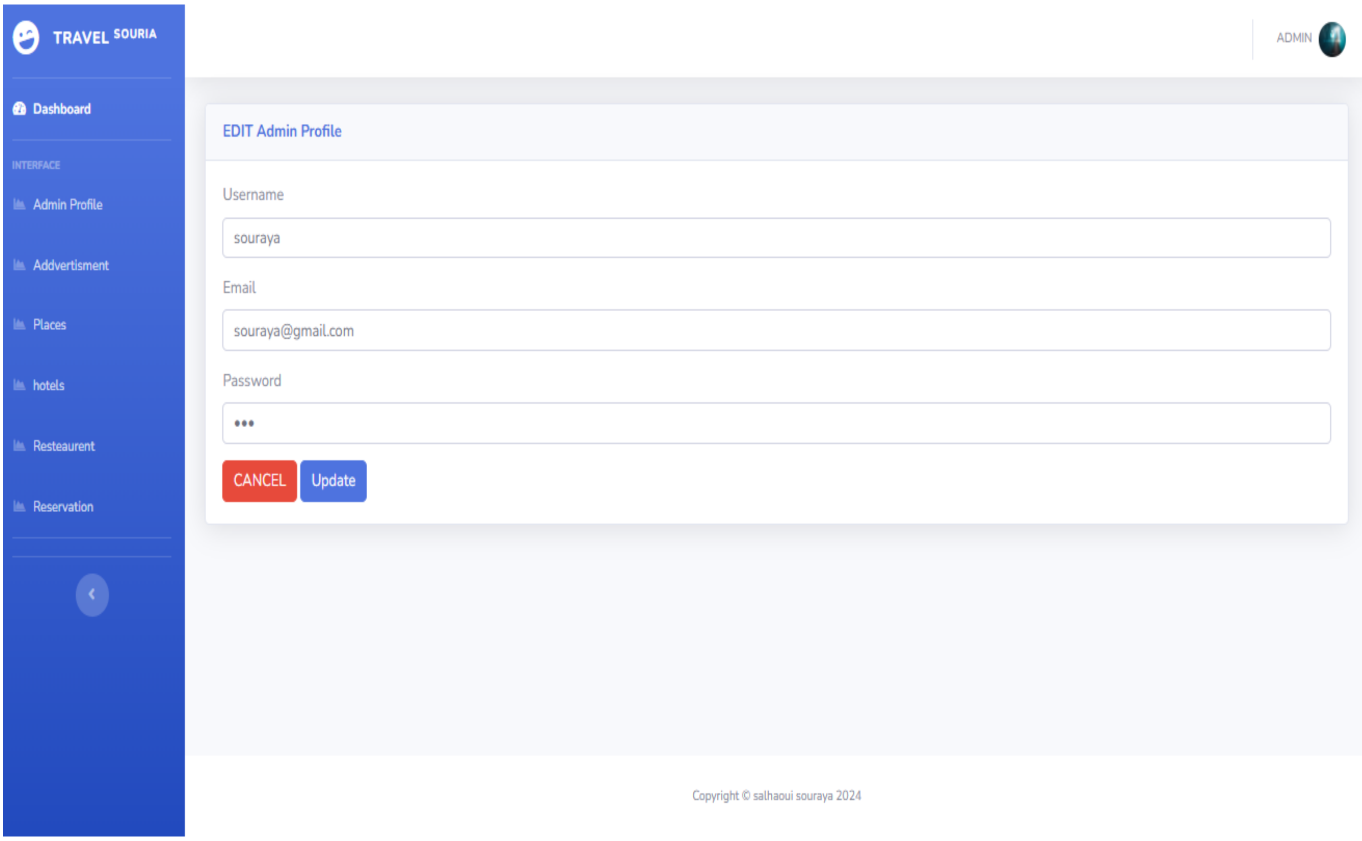


Figure 4.30: Update Admin Profile

4.3.2.3 Administrator settings Addvertisment :

In this part of the platform, advertisements containing discount offers are renewed by the administrator, and the details of each advertisement are as shown in the following image 4.31, in addition to the feature of deleting the advertisement when its duration expires and modifying the advertisement.

Also, new advertisements can be added using the Add Advertisement and Dictate button. Its information is as shown in image 4.32

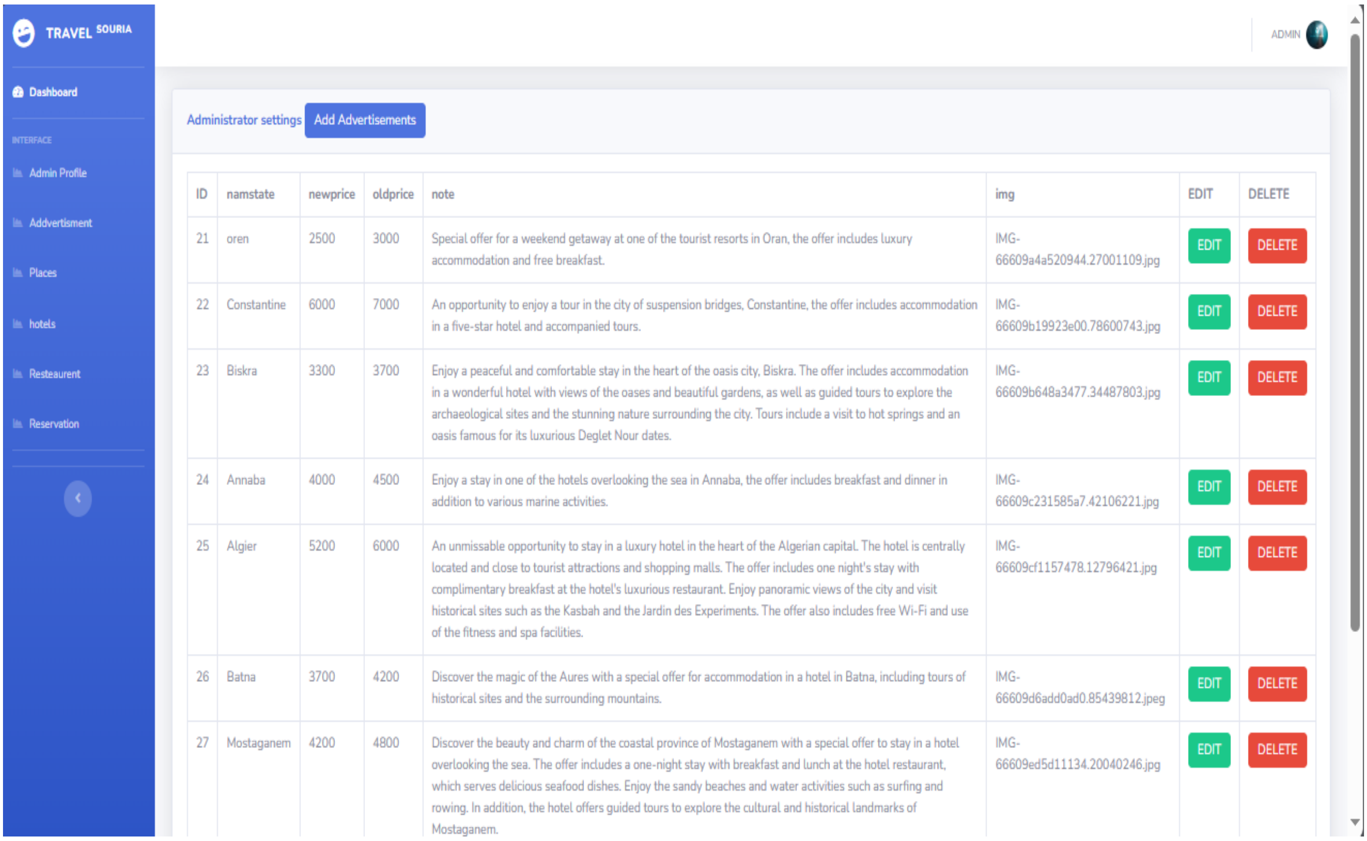


Figure 4.31: Addvertisment page

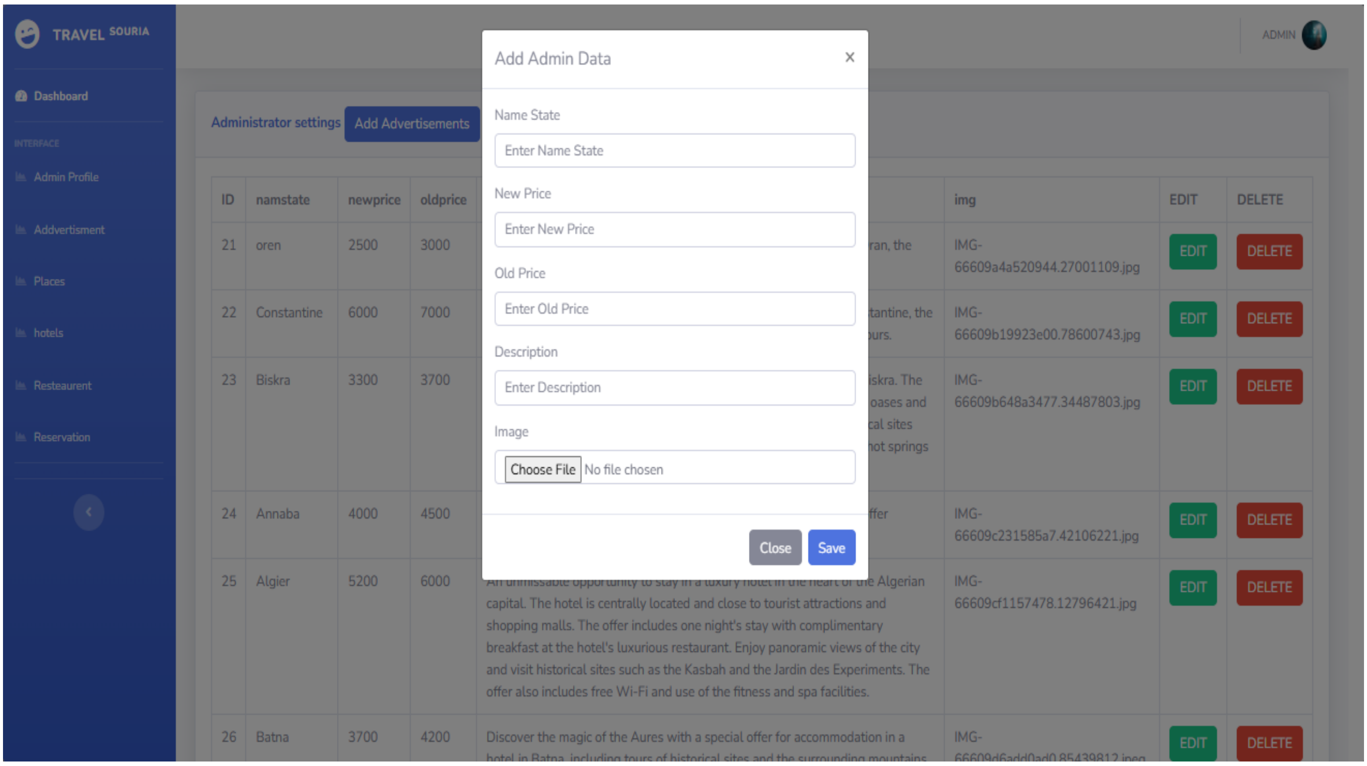


Figure 4.32: Add Addvertisment

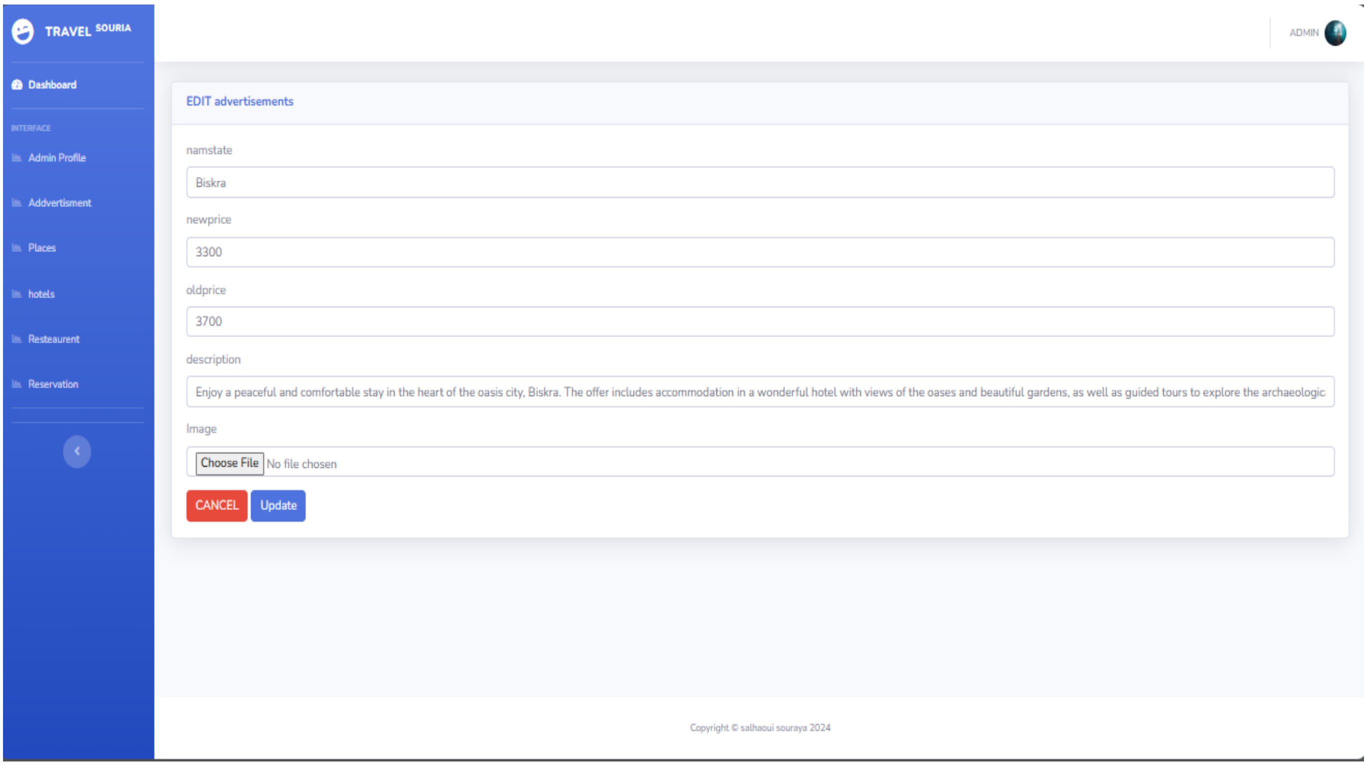


Figure 4.33: Update Addvertisment

4.3.2.4 Administrator settings Places:

We have allocated a special section for expanding data on places, so that each time new places are added and displayed to the user, we make the database large and diverse with places shown in Figure 4.34, where every administrator can add new places using the add button and fill out the information about the new place shown in Image 4.35 and Image 4.36 :

also can delete any area using the delete button or modify it using the edit button shown in Image

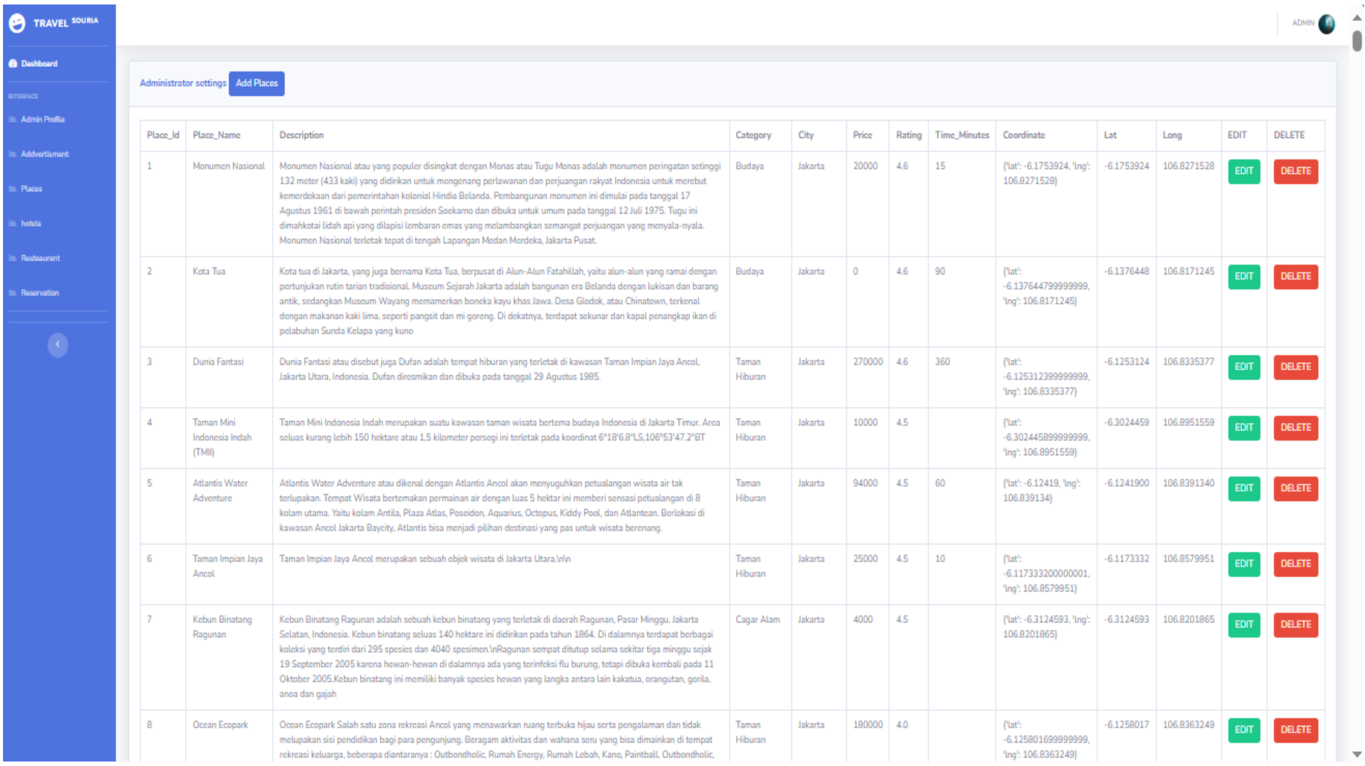
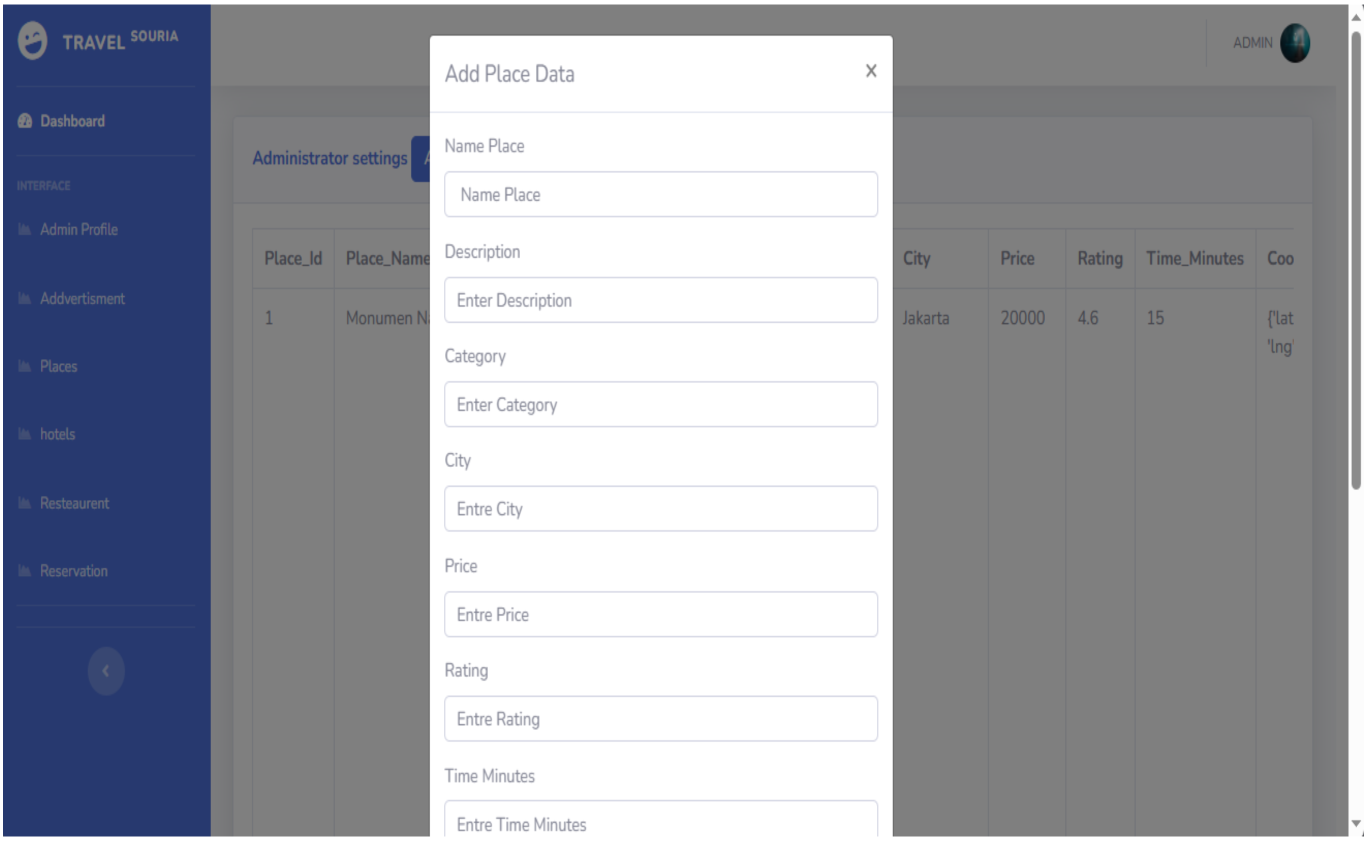


Figure 4.34: Places Page



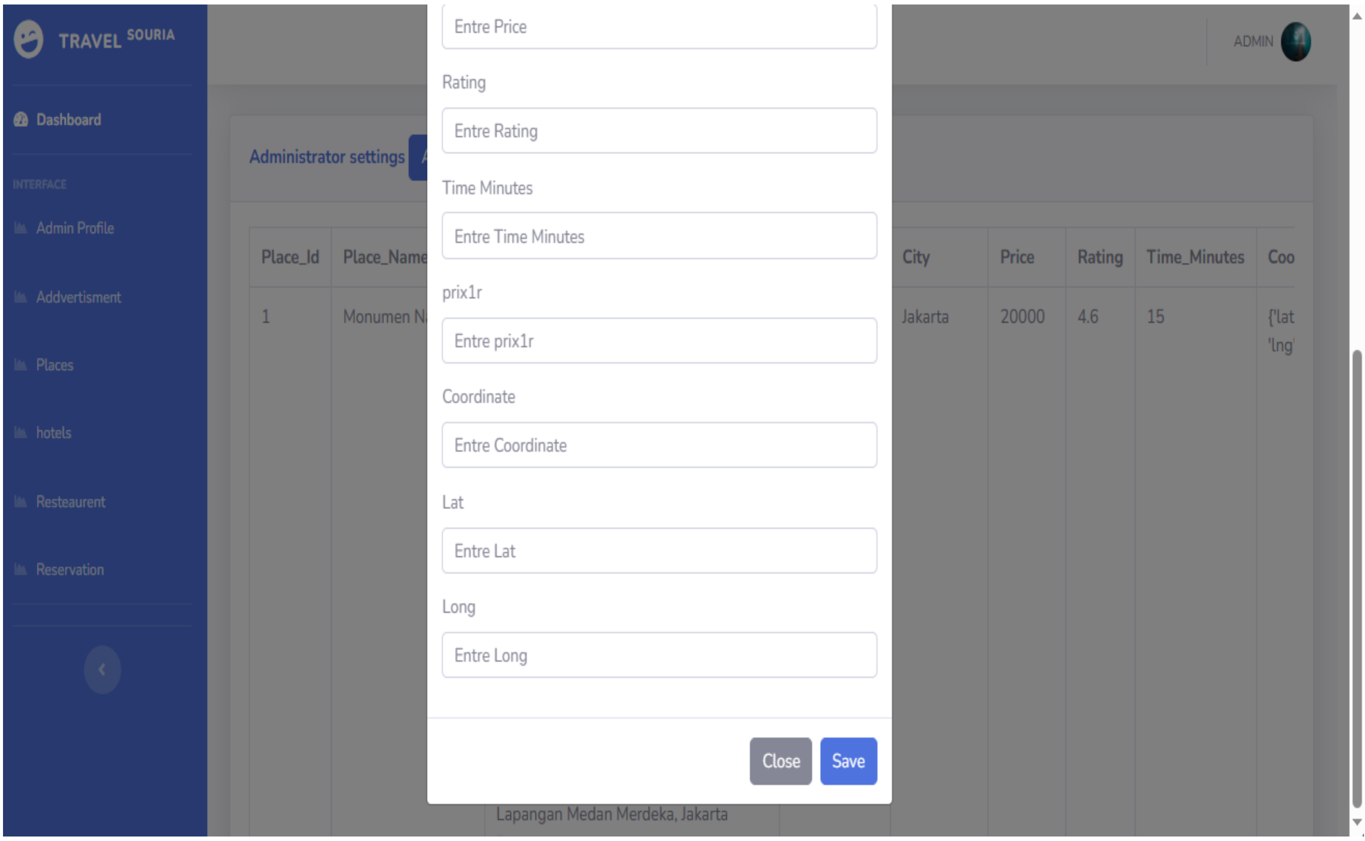


Figure 4.35: Add Places

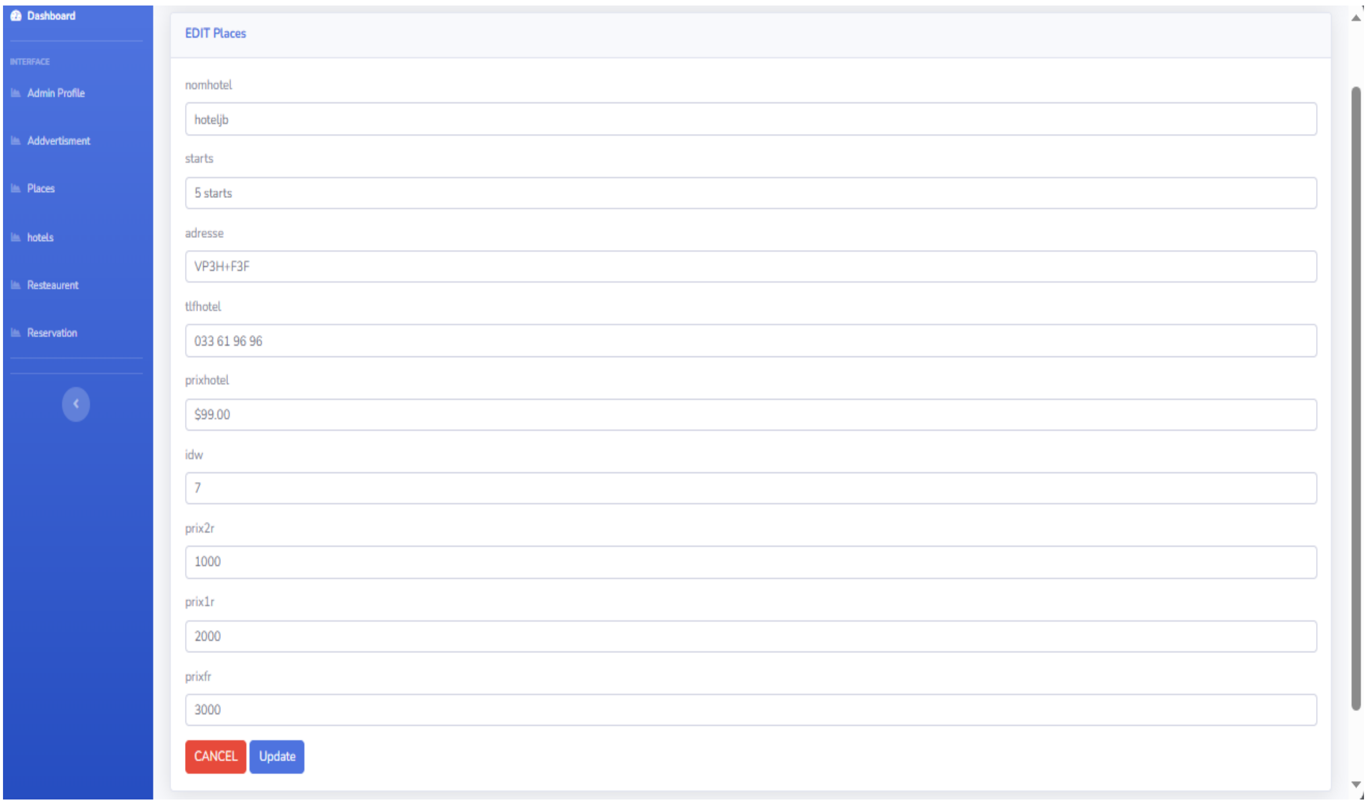
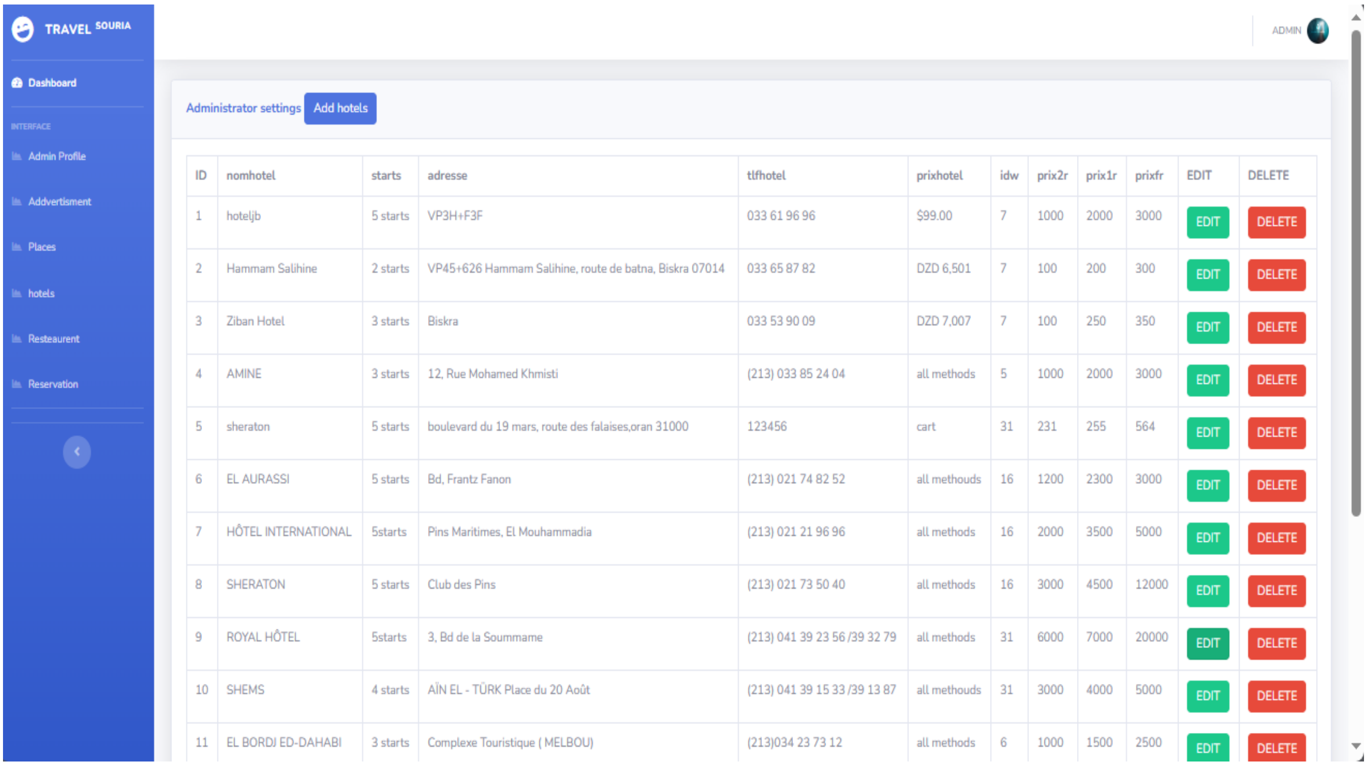


Figure 4.36: Update Places

4.3.2.5 Administrator settings Hotels:

We clearly see in image 4.37 a table displaying all the hotels in the database displayed by the application

To make the database large, this is a section dedicated to adding hotels by administrators. This is done through the Add Hotels button and fill out the displayed information as shown in Image 4.38 It is then displayed in the application to the user where there is a deletion feature, and the information can also be modified as shown in Picture 4.39



ID	nomhotel	starts	adresse	tlfhotel	prixhotel	idw	prix2r	prix1r	prixfr	EDIT	DELETE
1	hoteljb	5 starts	VP3H+F3F	033 61 96 96	$99.00	7	1000	2000	3000	EDIT	DELETE
2	Hammam Salihine	2 starts	VP45+626 Hammam Salihine, route de batna, Biskra 07014	033 65 87 82	DZD 6,501	7	100	200	300	EDIT	DELETE
3	Ziban Hotel	3 starts	Biskra	033 53 90 09	DZD 7,007	7	100	250	350	EDIT	DELETE
4	AMINE	3 starts	12, Rue Mohamed Khmisti	(213) 033 85 24 04	all methods	5	1000	2000	3000	EDIT	DELETE
5	sheraton	5 starts	boulevard du 19 mars, route des falaises,oran 31000	123456	cart	31	231	255	564	EDIT	DELETE
6	EL AURASSI	5 starts	Bd, Frantz Fanon	(213) 021 74 82 52	all methouds	16	1200	2300	3000	EDIT	DELETE
7	HÔTEL INTERNATIONAL	5starts	Pins Maritimes, El Mouhammadia	(213) 021 21 96 96	all methods	16	2000	3500	5000	EDIT	DELETE
8	SHERATON	5 starts	Club des Pins	(213) 021 73 50 40	all methods	16	3000	4500	12000	EDIT	DELETE
9	ROYAL HÔTEL	5starts	3, Bd de la Soummame	(213) 041 39 23 56 /39 32 79	all methods	31	6000	7000	20000	EDIT	DELETE
10	SHEMS	4 starts	AÏN EL - TÜRK Place du 20 Août	(213) 041 39 15 33 /39 13 87	all methouds	31	3000	4000	5000	EDIT	DELETE
11	EL BORDJ ED-DAHABI	3 starts	Complexe Touristique (MELBOU)	(213)034 23 73 12	all methods	6	1000	1500	2500	EDIT	DELETE

Figure 4.37: Hotels page

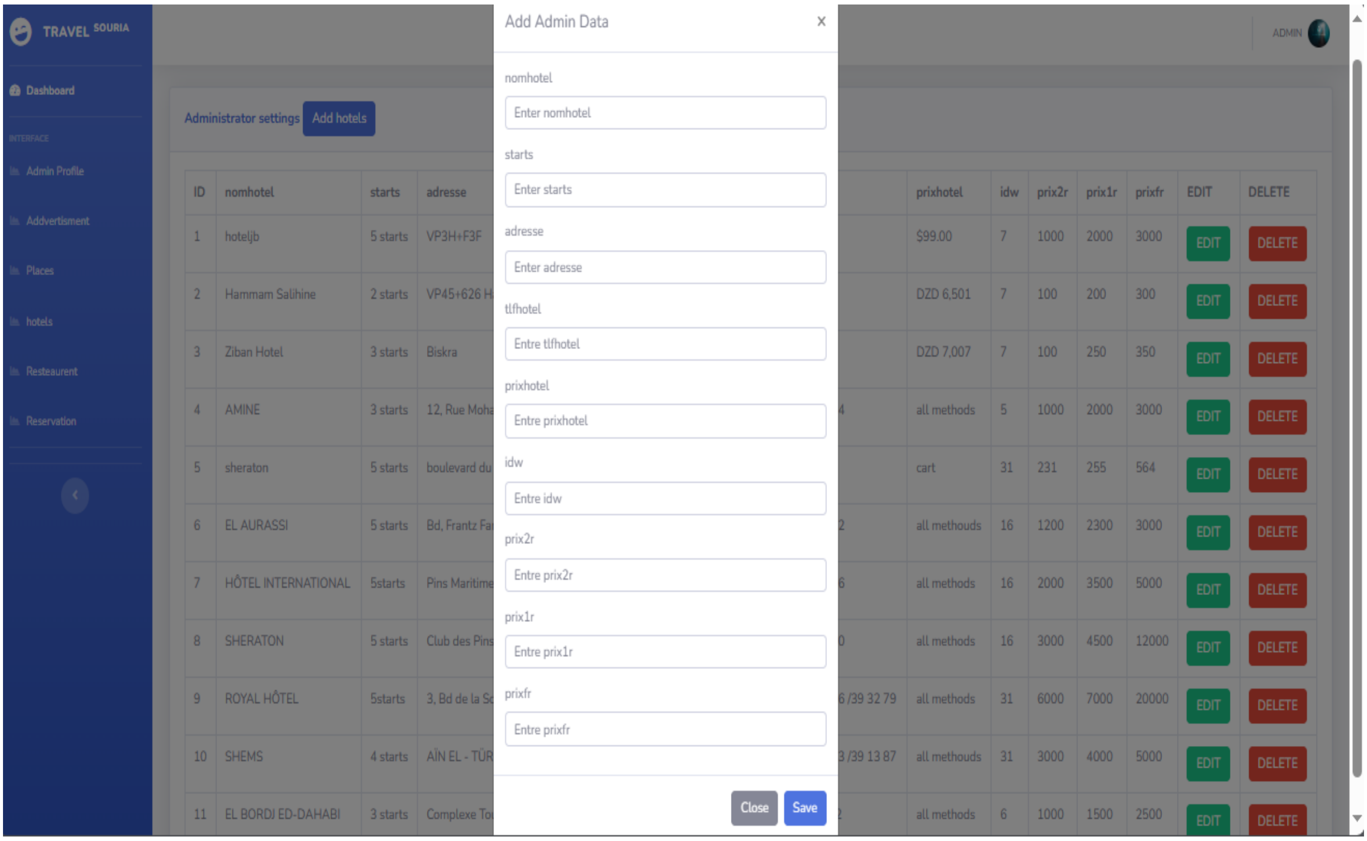


Figure 4.38: Add Hotels

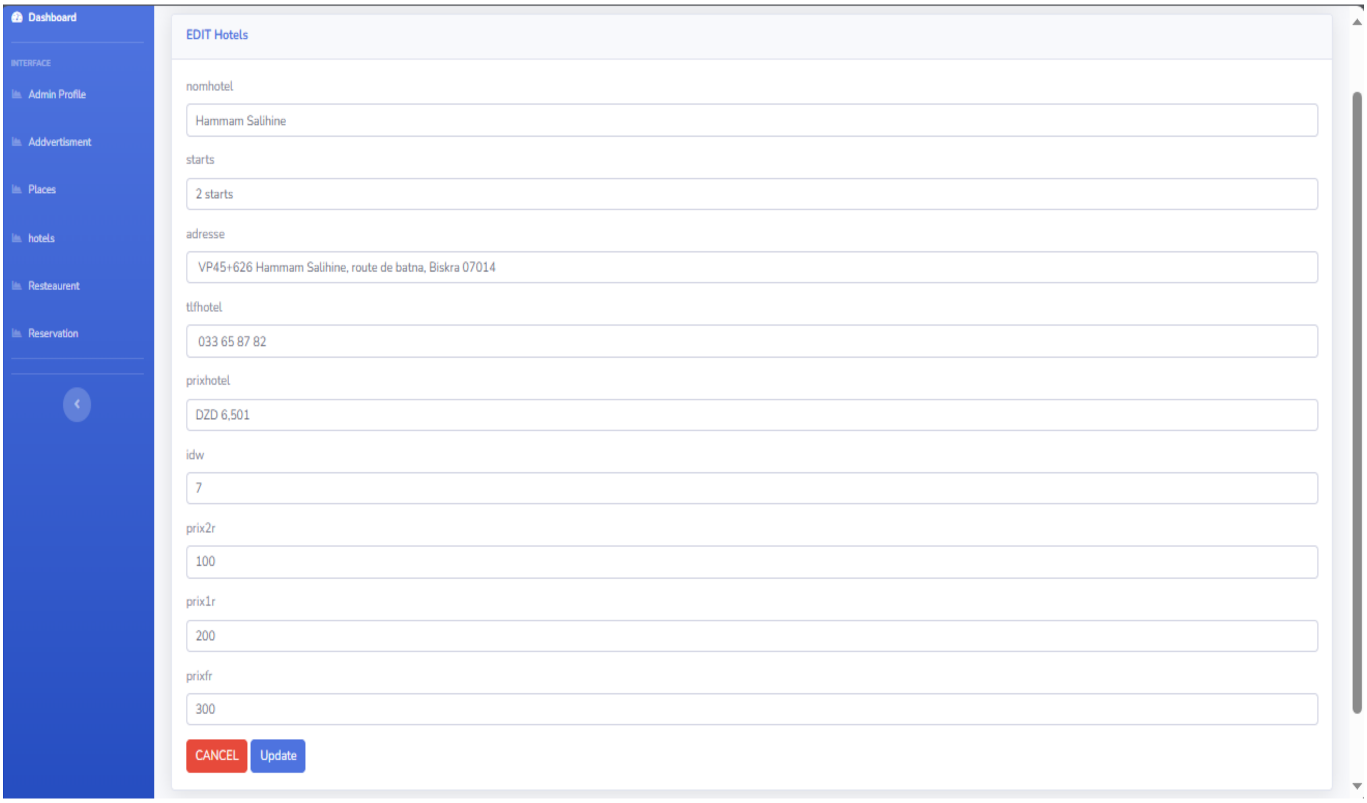


Figure 4.39: Update Hotels

4.3.2.6 Administrator settings Restaurent:

Restaurant service has also been allocated where the administrator can Because restaurants are also among the most important and priorities of tourists, a special service has been allocated for them, shown in Picture 4.40, where all the restaurants located in various places are added on a regular basis, filling out all information related to them, as shown in Picture 4.41, and all information can be renewed in the event of Changing one of the pieces of information is shown as follows in Image 4.42, as well as the deletion process .

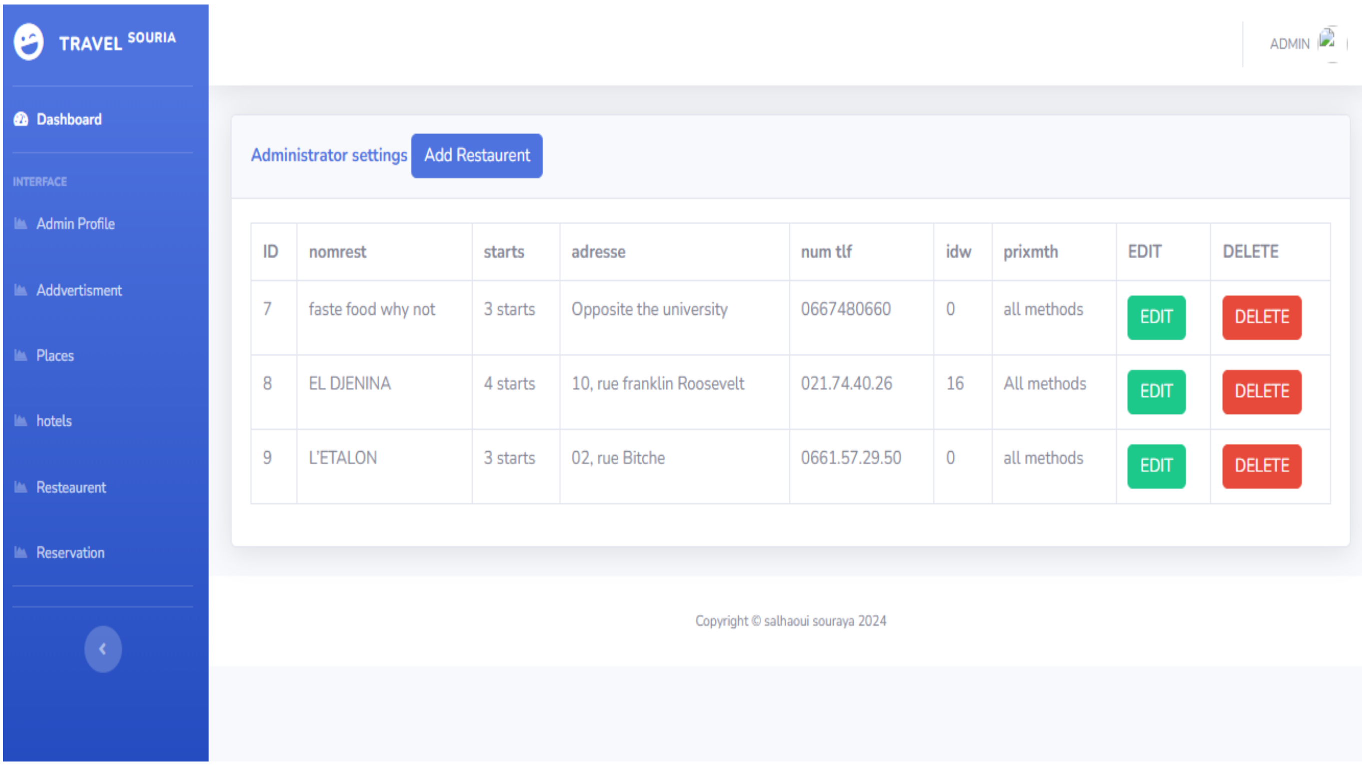


Figure 4.40: Restaurent Page

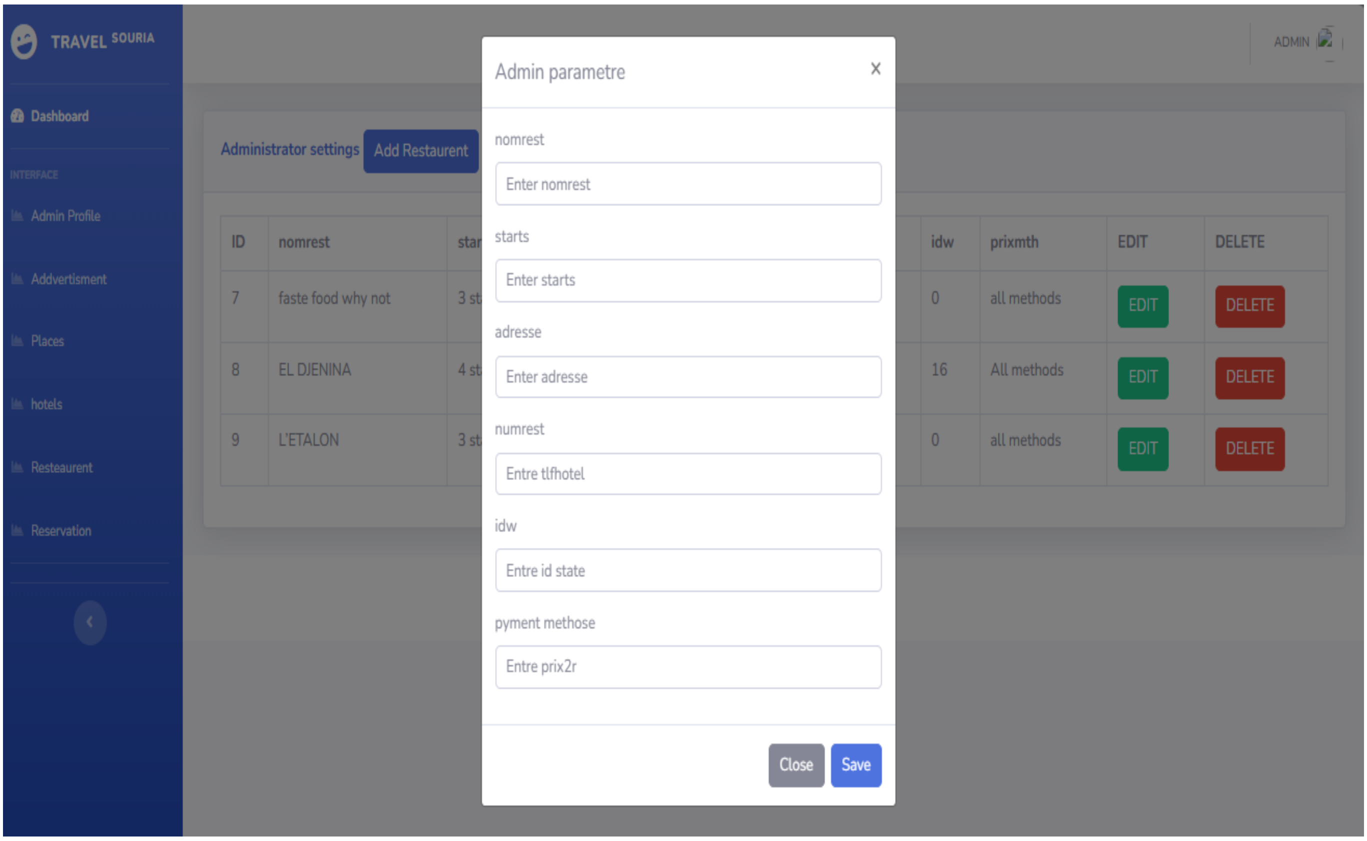


Figure 4.41: Add Restaurent

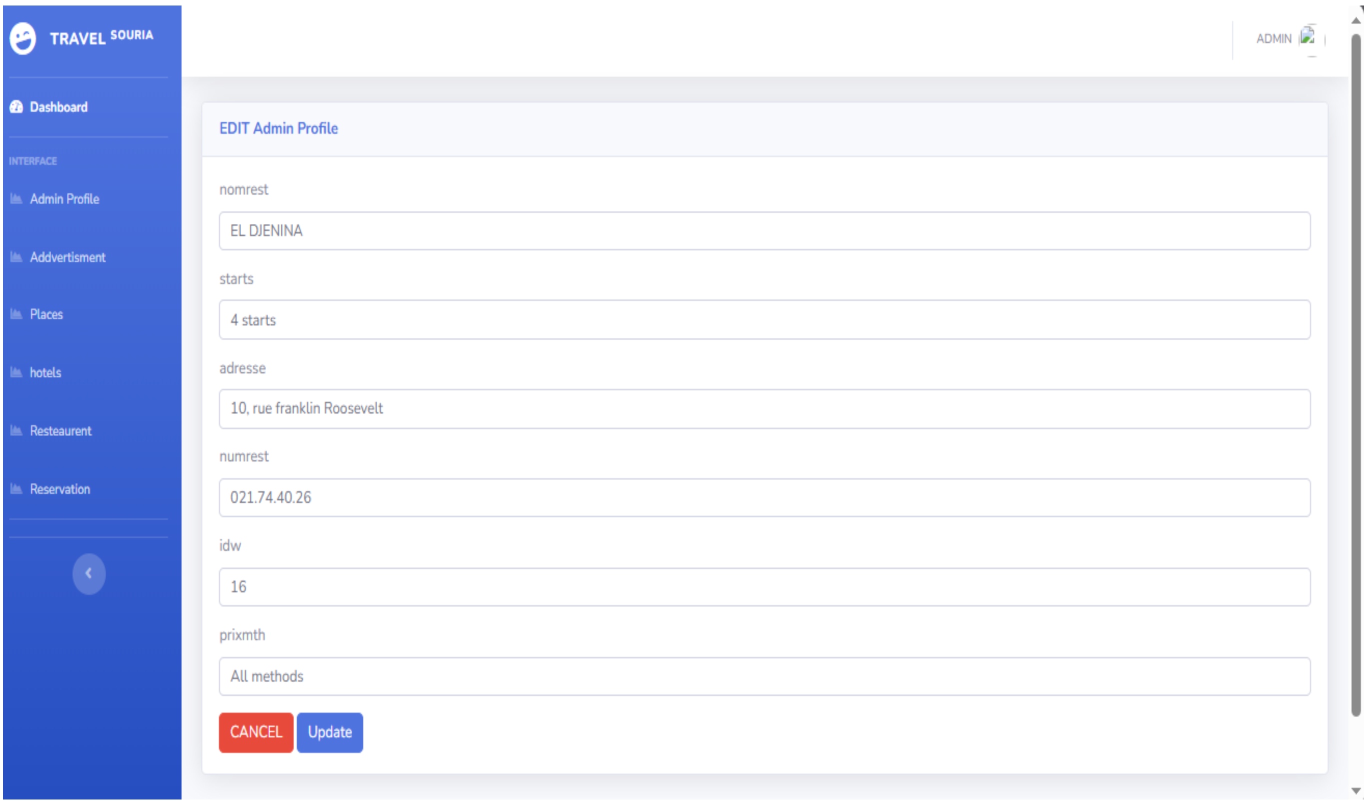


Figure 4.42: Update Restaurent

4.3.2.7 Administrator settings Reservation:

In this reservations section, a list of all people who have made a reservation will appear to the administrator.as shown in image 4.43

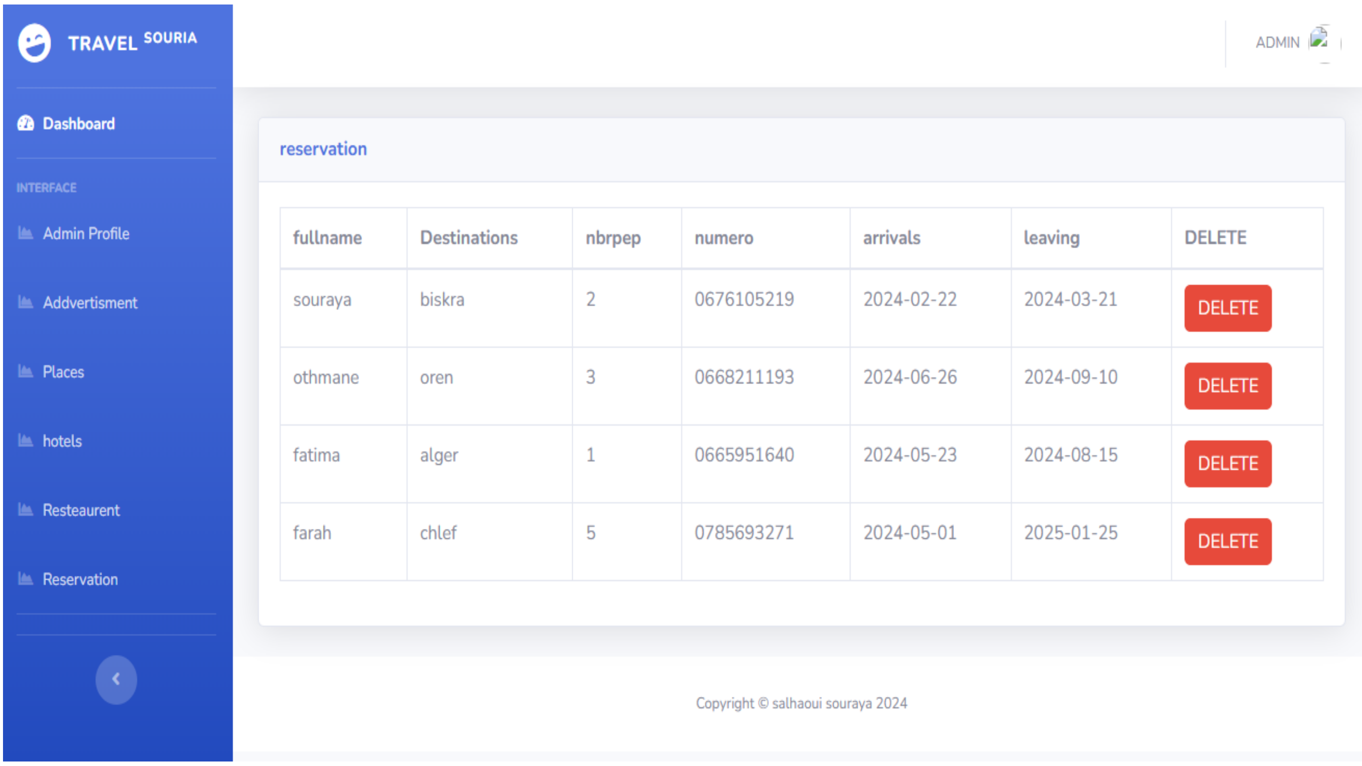


Figure 4.43: Liste of Reservation

4.3.3 user

4.3.3.1 Rating

On the user's page, the section for liking appears in the Dashboard interface, as shown in image 4.44 , Image 4.45 shows the stages of adding a like, which is in the name of the user, the location that liked it, and its star rating.

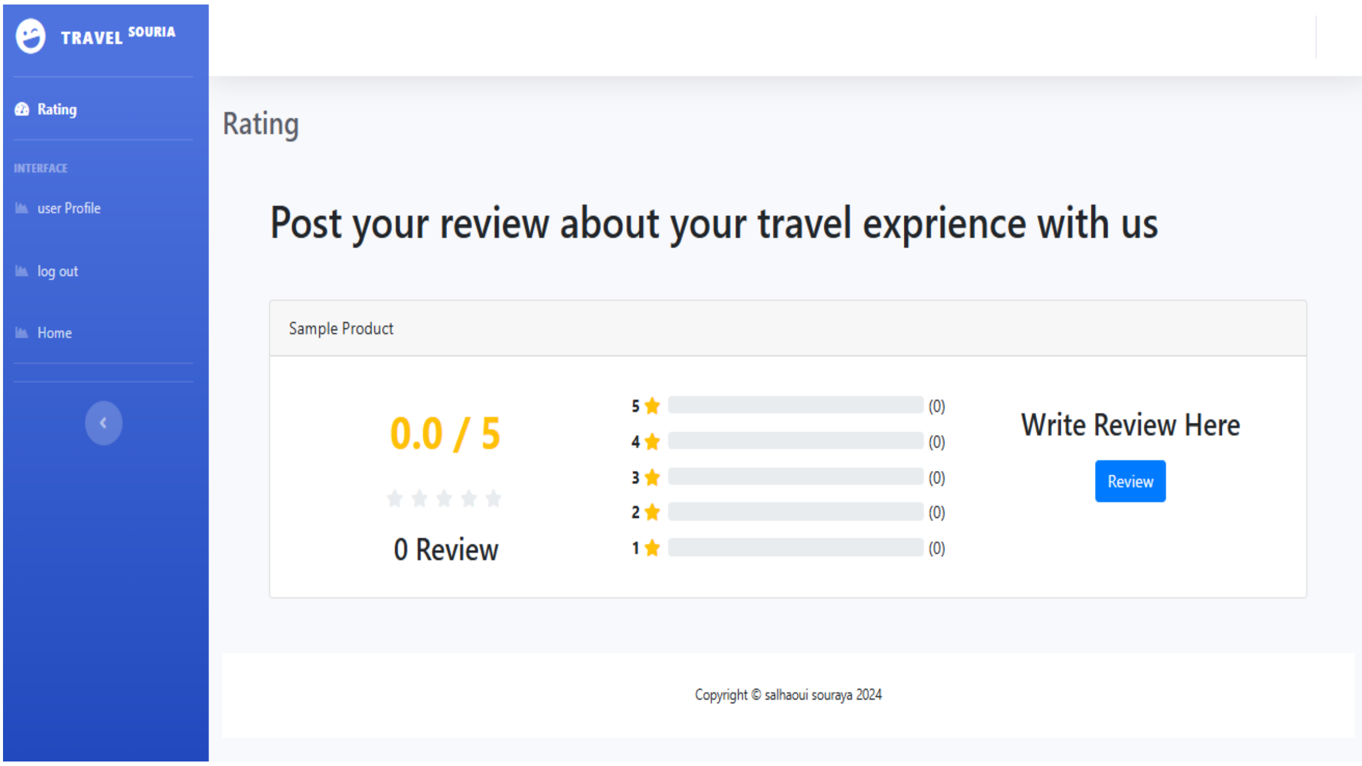


Figure 4.44: Rating page

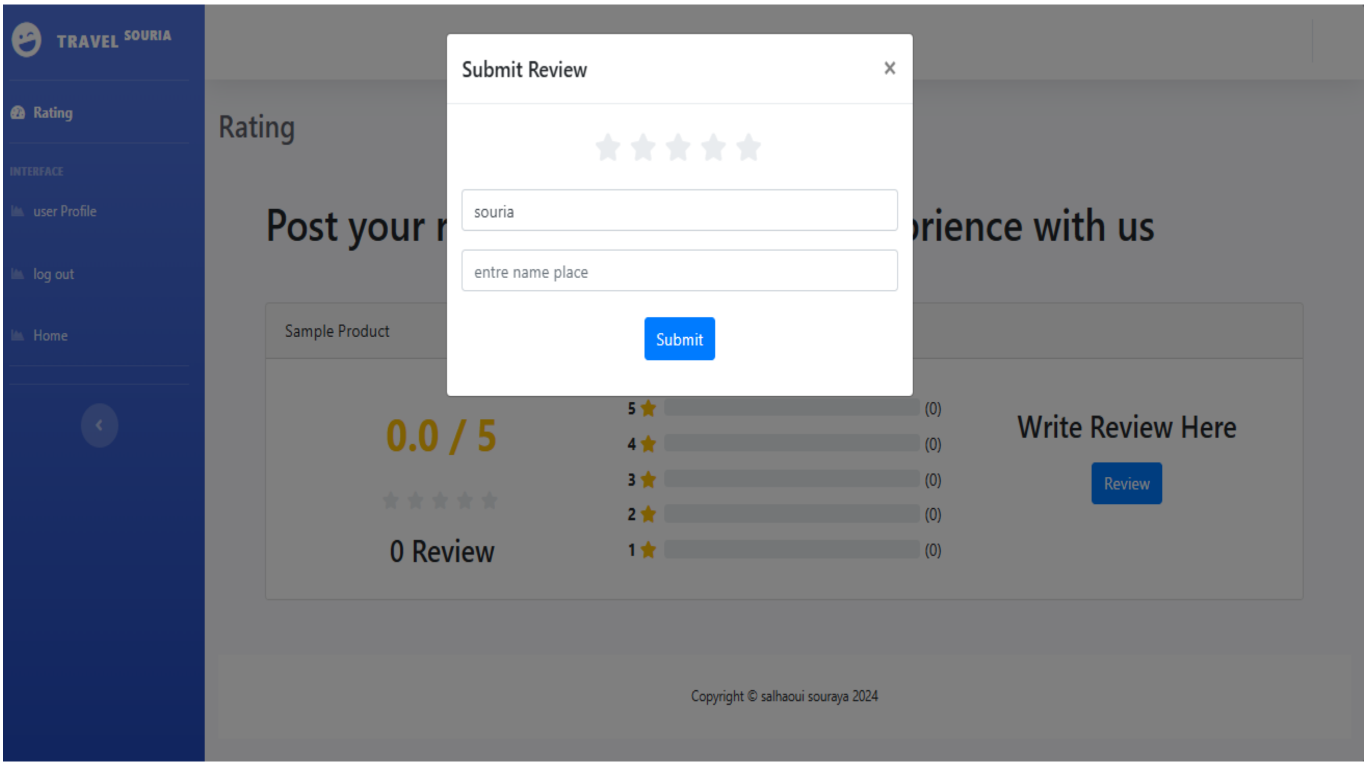


Figure 4.45: Add Rating

4.3.3.2 EDIT user Profile

On page 4.46, the modifications that the user can make appear. He should do it on his personal file that contains his information

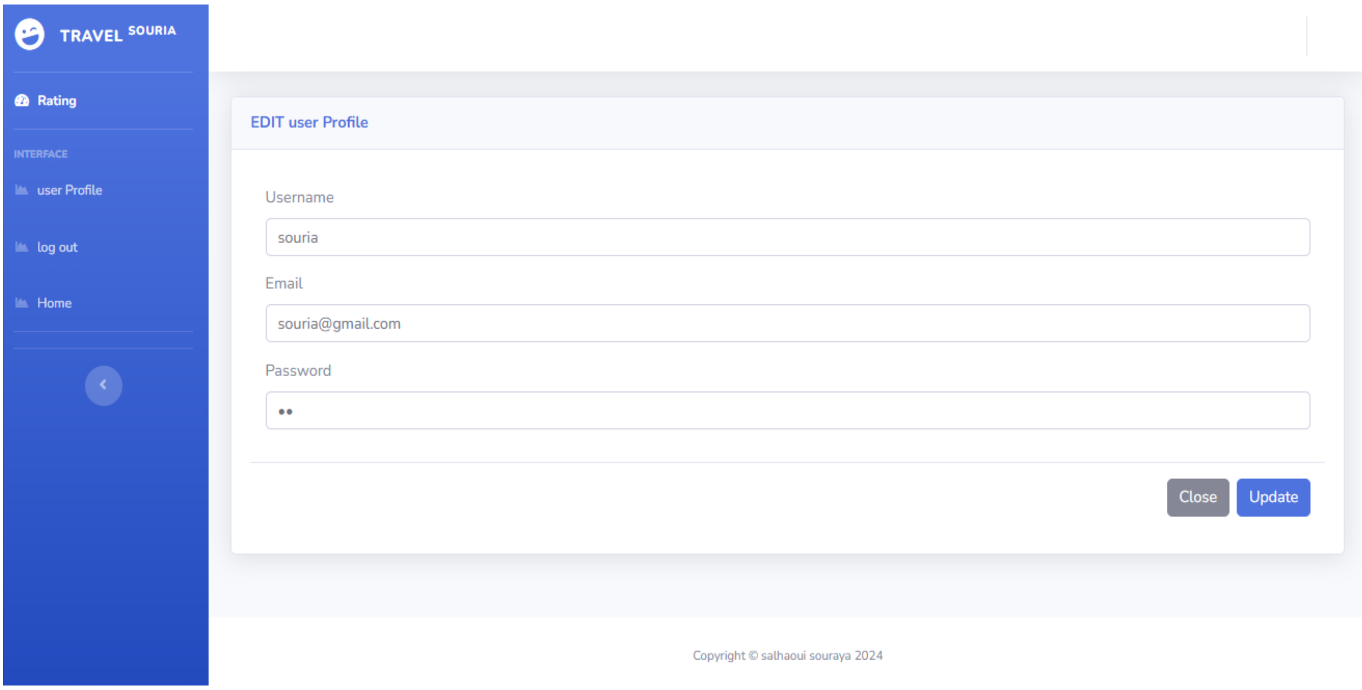


Figure 4.46: EDIT user Profile

4.3.4 home

Figure 4.47 describe our web Application home interface where :

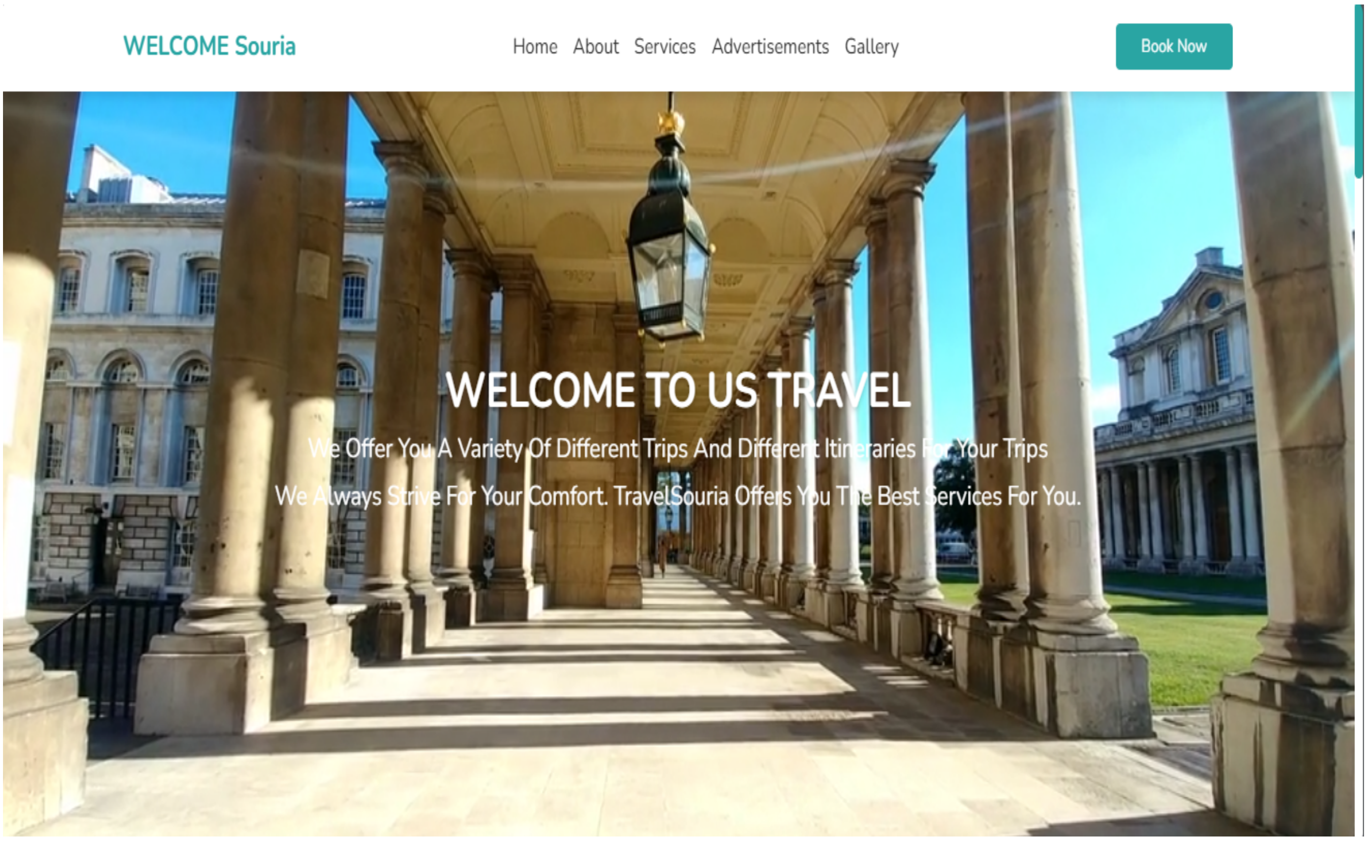


Figure 4.47: Home Page

1. **About :** Description About us and our Start-up. as shown in the image 4.48

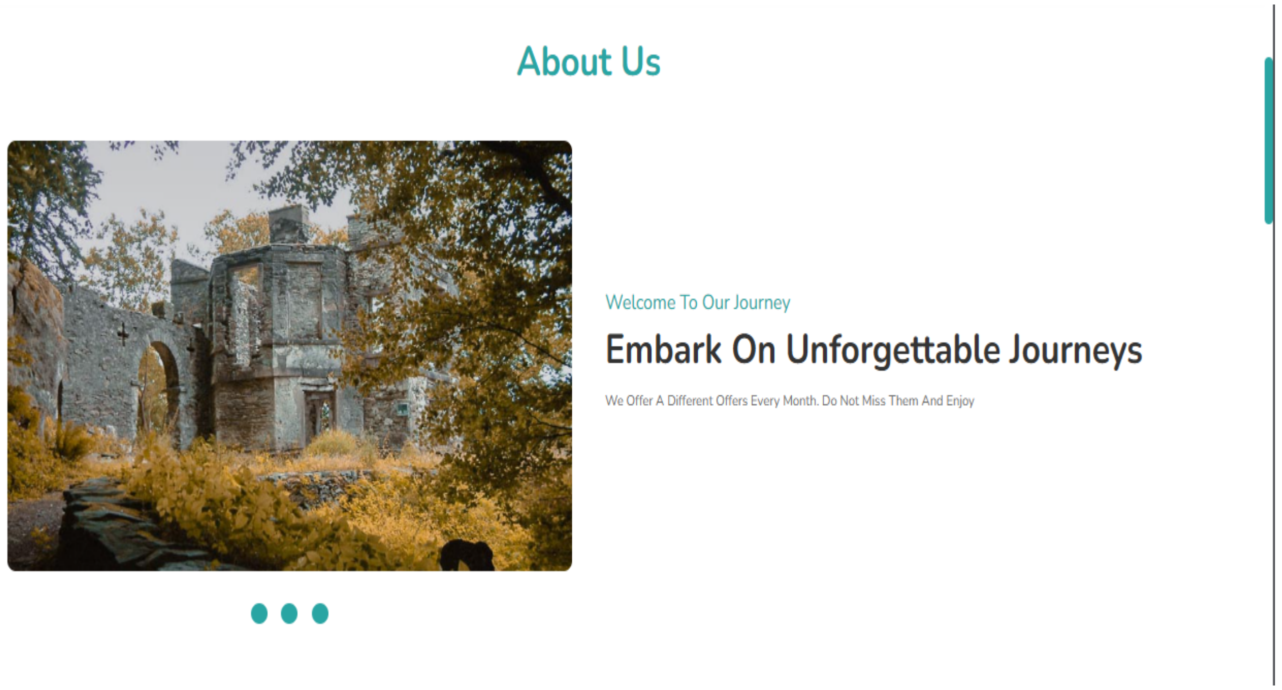


Figure 4.48: About Page

2. **Services :** Describes the various sub-services available in our web application . as shown in the image 4.49

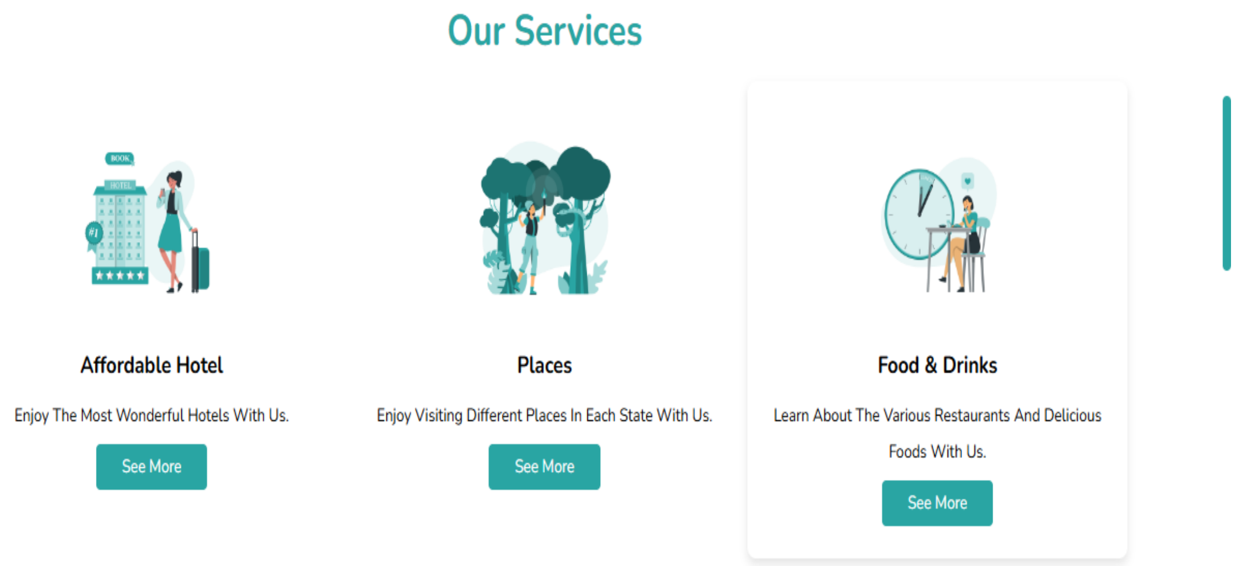


Figure 4.49: Services Page

3. **Advertisements :** A special section for the organization's advertisements and offers, which include discounts on trips. as shown in the image 4.50 and image 4.51

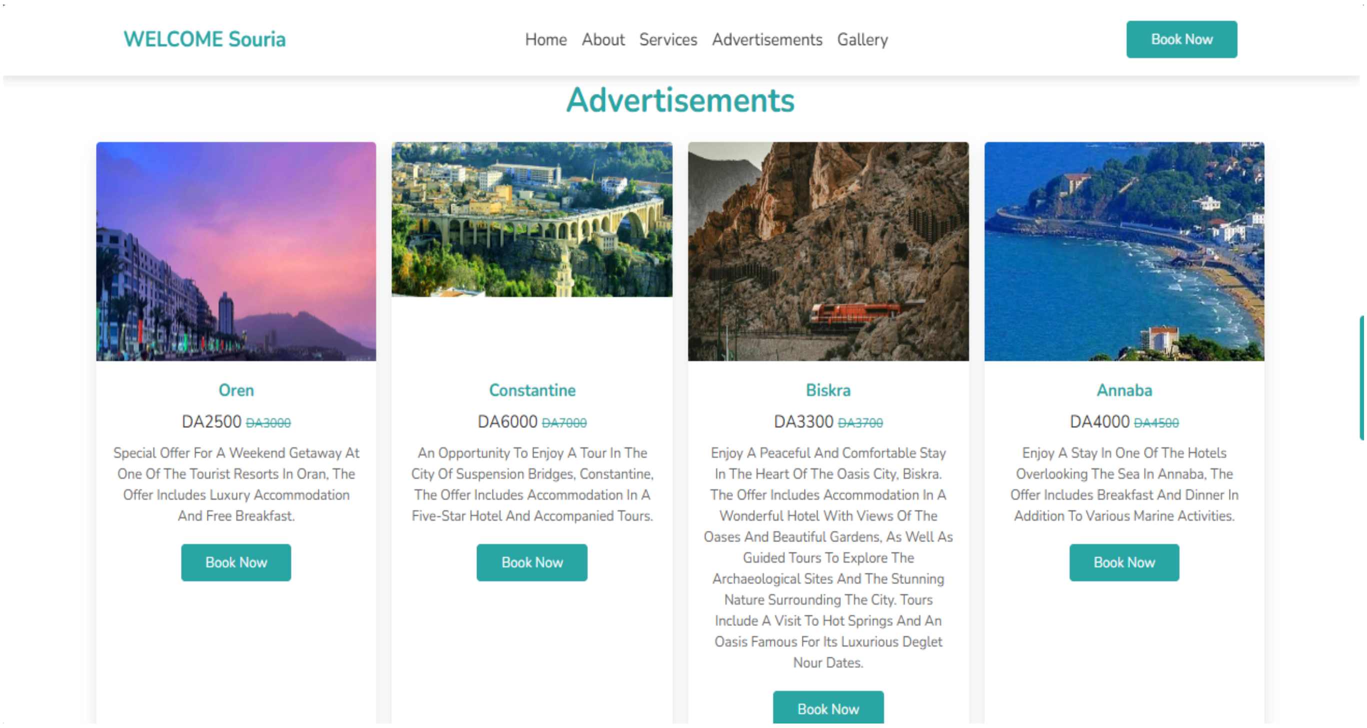


Figure 4.50: Advertisements Page -1-

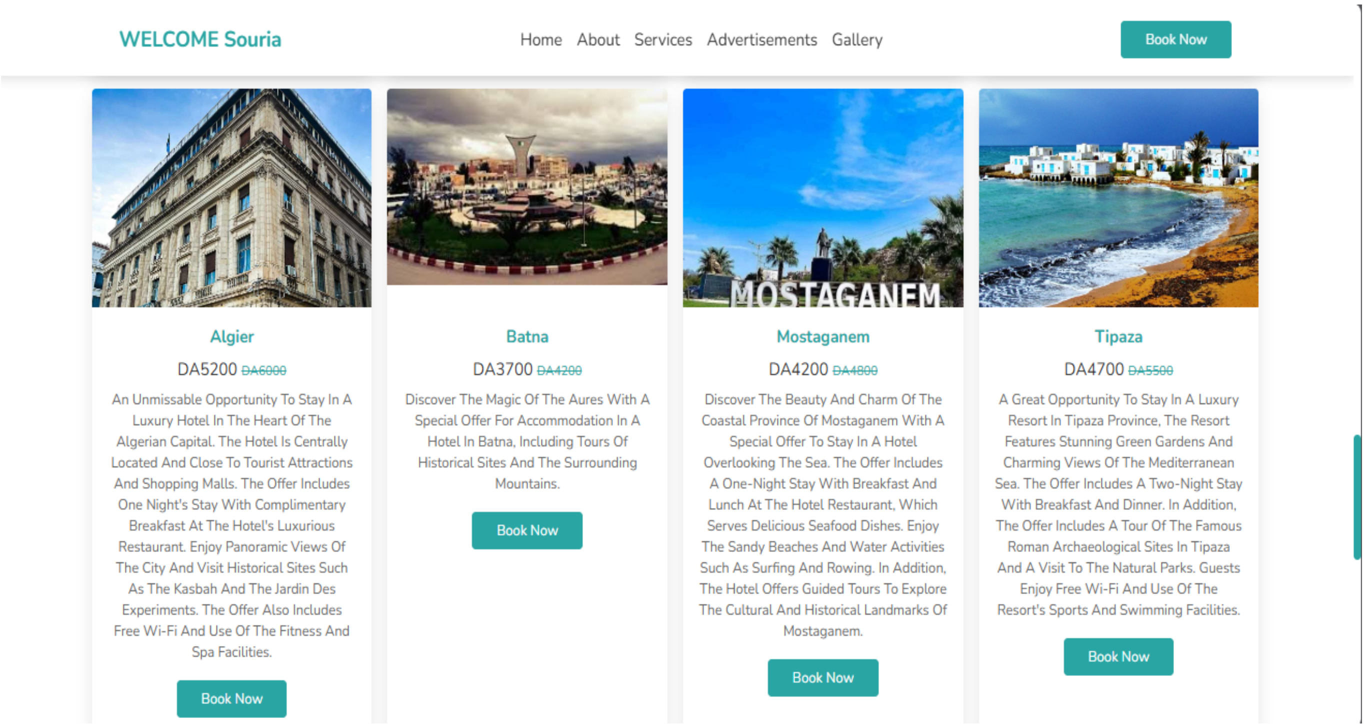


Figure 4.51: Advertisements Page -2-

4. **Gallery :** It is a photo gallery of the various states in our country, as well as giving tourists an overview of the beautiful areas . as shown in the image 4.52

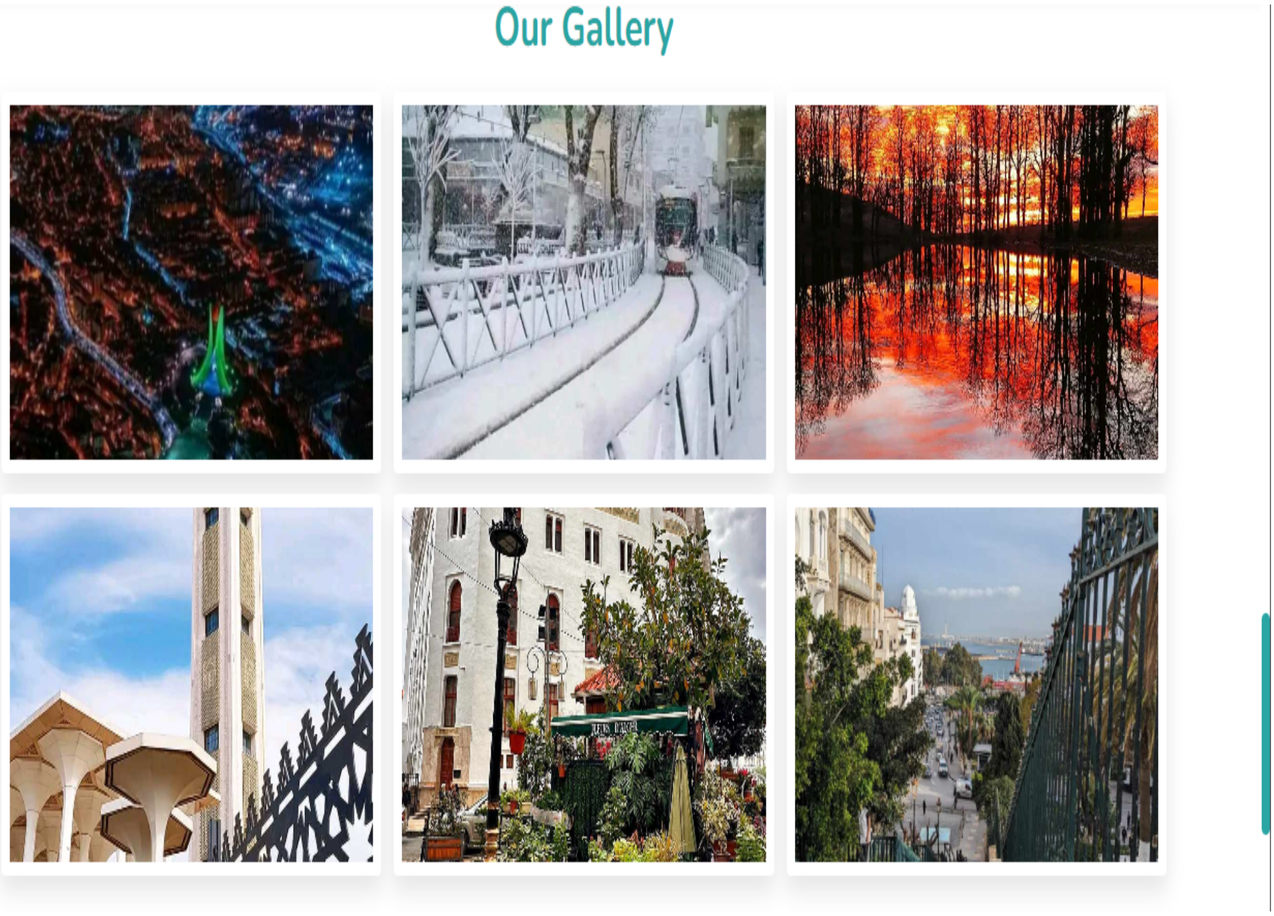


Figure 4.52: Gallery Page

5. **information :**At the end of the page there is information related to the web application for anyone who wants to contact us and for more information. As shown in picture 4.53

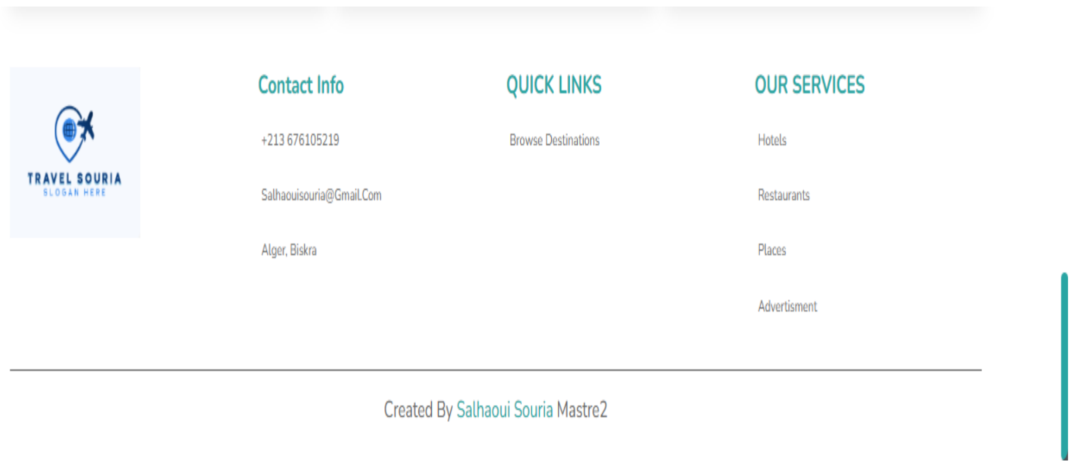


Figure 4.53: information Page

6. **book :** Any user who wants to book can click on the button for booking flights shown in the following image 4.54 and in the image 4.55 the Reply book.

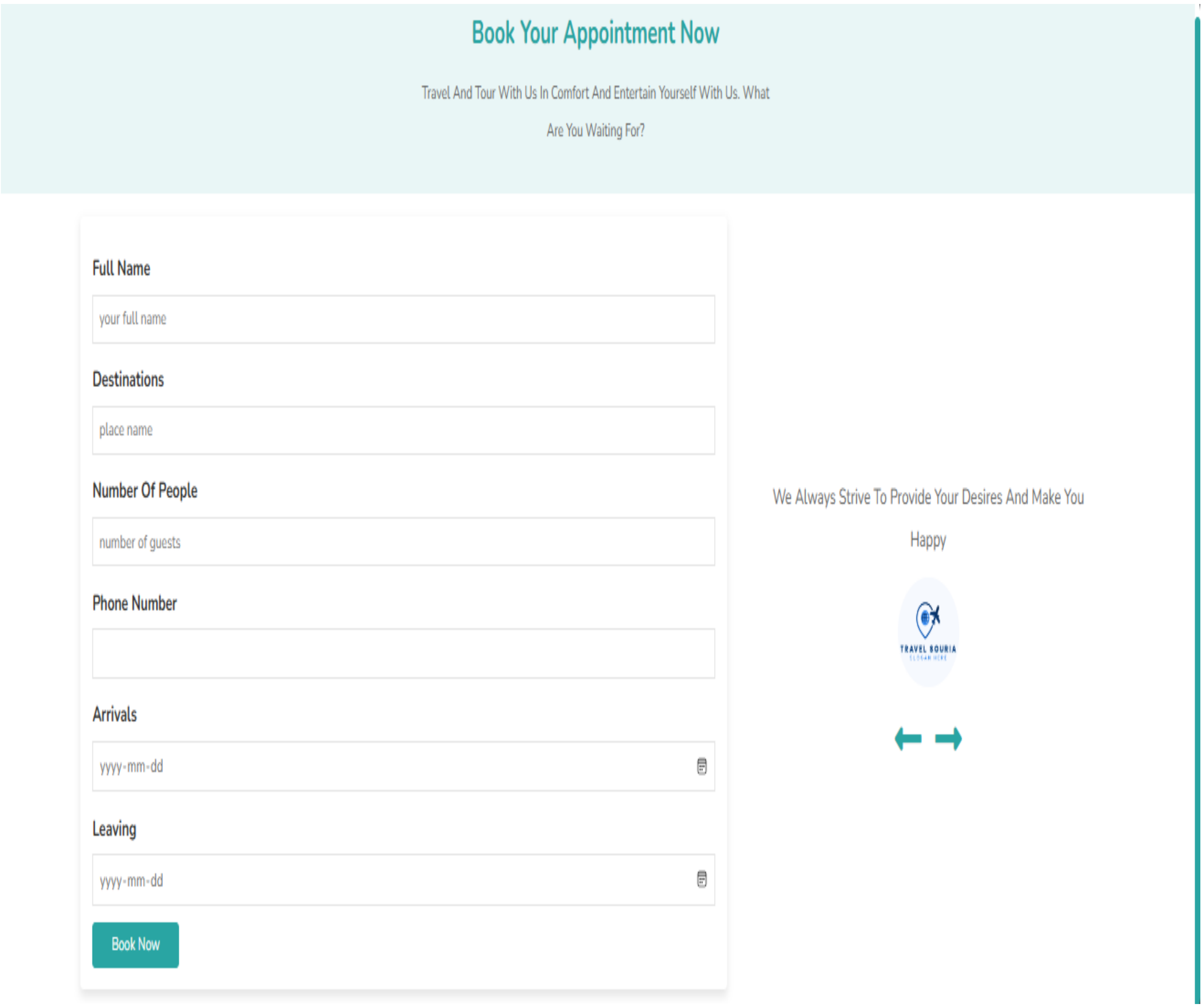


Figure 4.54: book Page

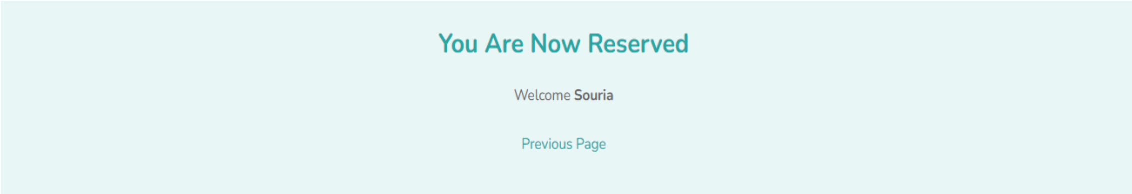


Figure 4.55: Reply book

4.3.5 Hotels Services

Figure 4.56 describe our web application Hotels Services home interface where : In this part, the user tests the state in which he wants to search for hotels, and by selecting it, a list will automatically appear to him containing all the hotels it contains. After that, the user clicks on the hotel he desires

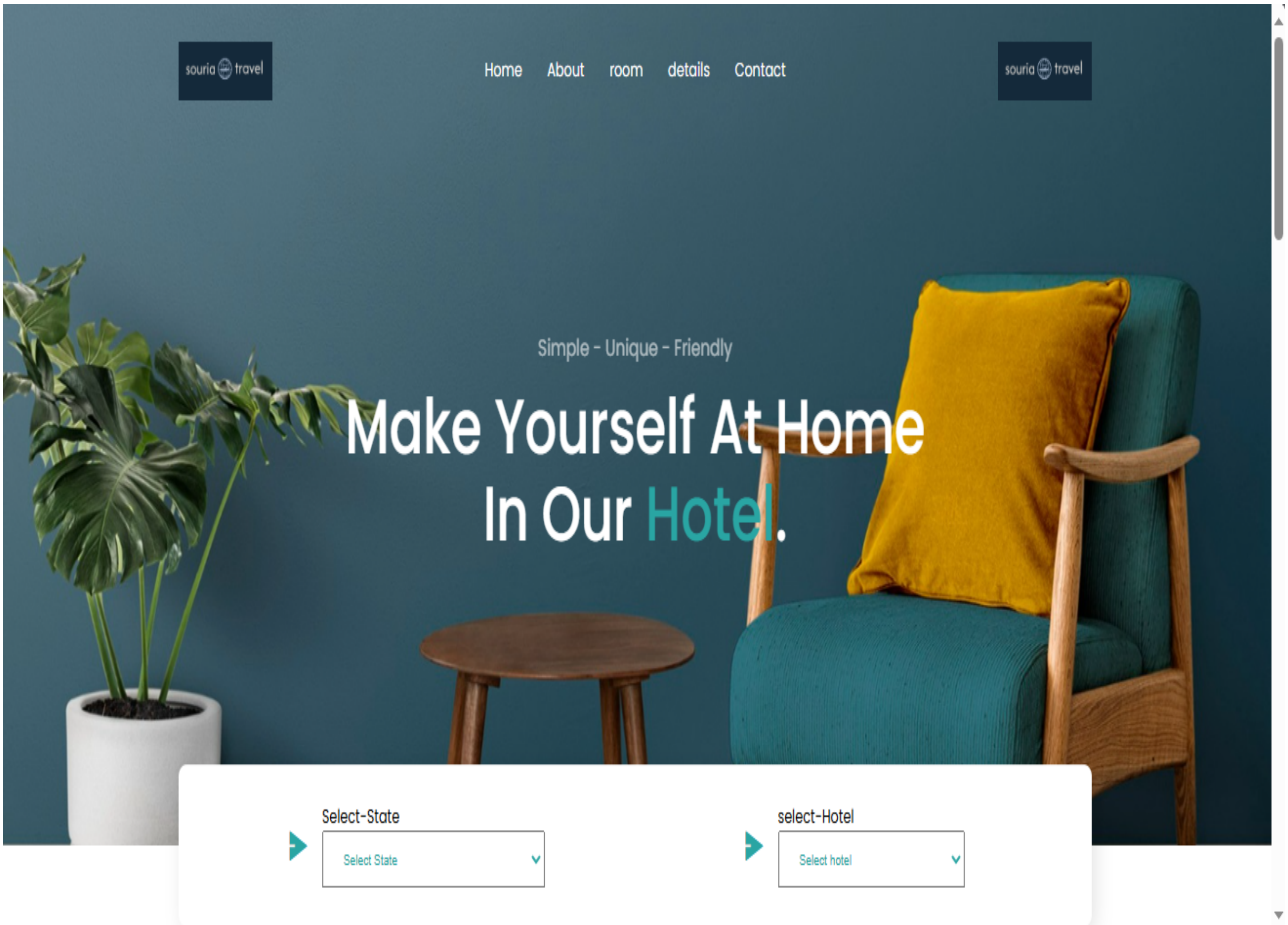


Figure 4.56: Home Hotel service

1. **About :** Providing a short Description about hotels and their information .As shown in picture 4.57

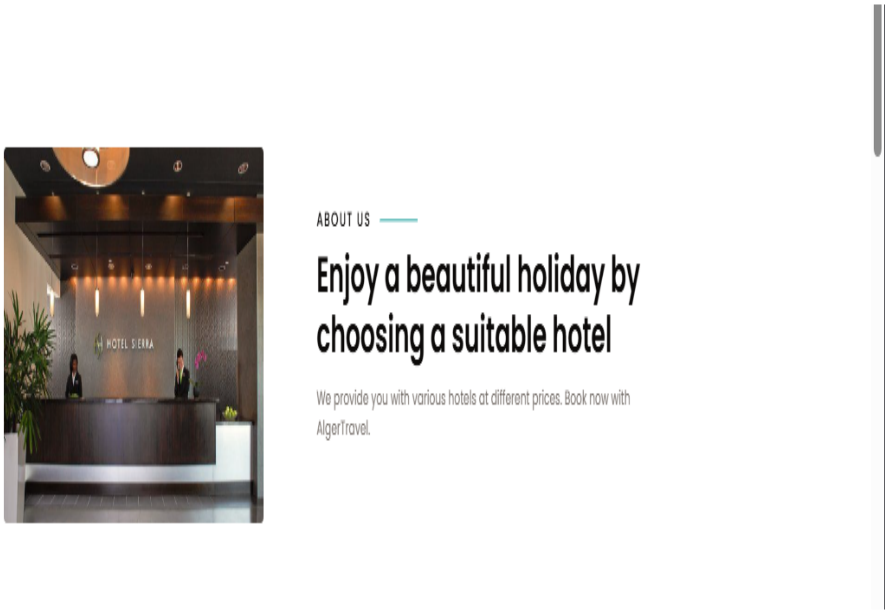


Figure 4.57: About Hotel service

2. **room :** When the user click on the hotel you want, information about the prices for the double room, single room, and family room appears in this section.As shown in picture 4.58

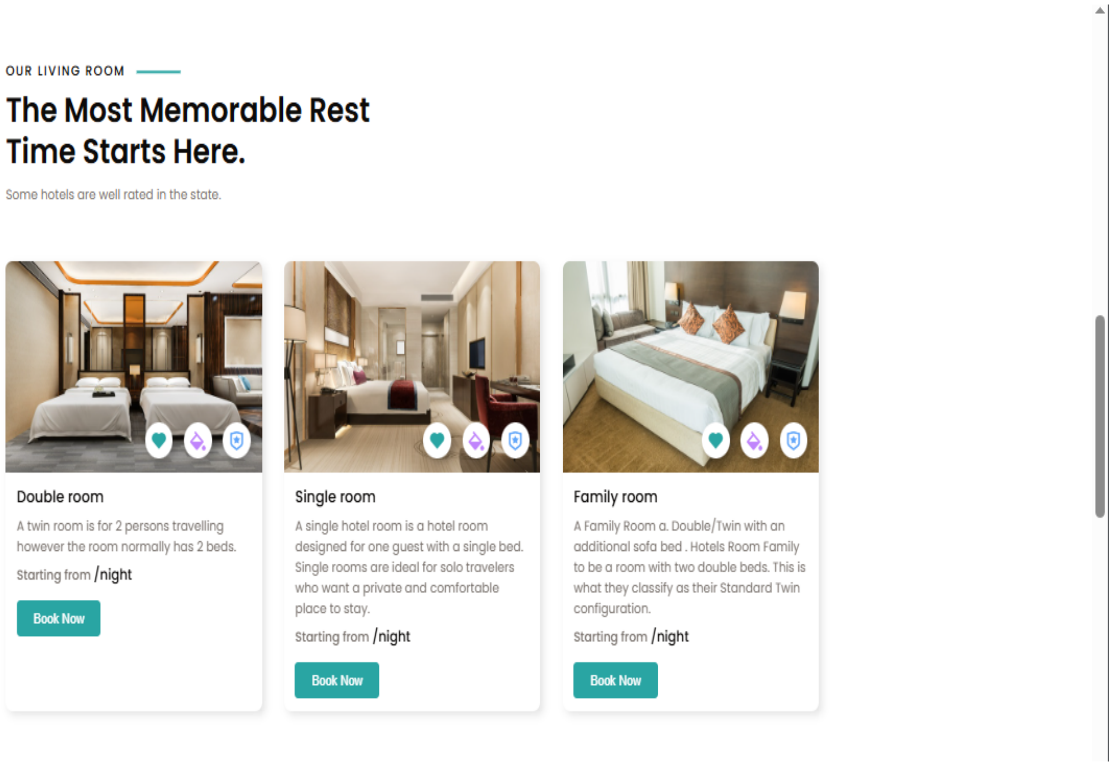


Figure 4.58: room Hotel service

3. **details :** When the user click on the selected hotel he wants, the detailed information for the hotel appears in this part, which is shown in the following image 4.59

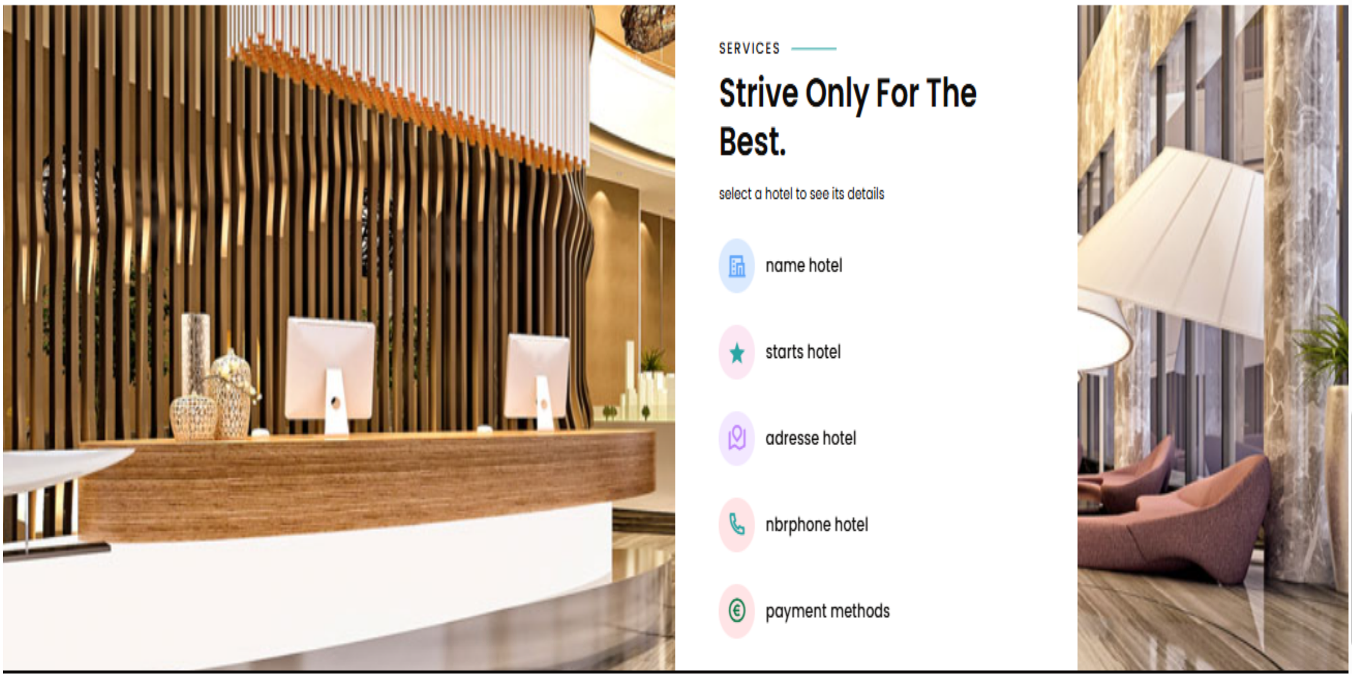


Figure 4.59: details Hotel service

4. **Contact :**Our information to contact us . As shown in picture 4.60

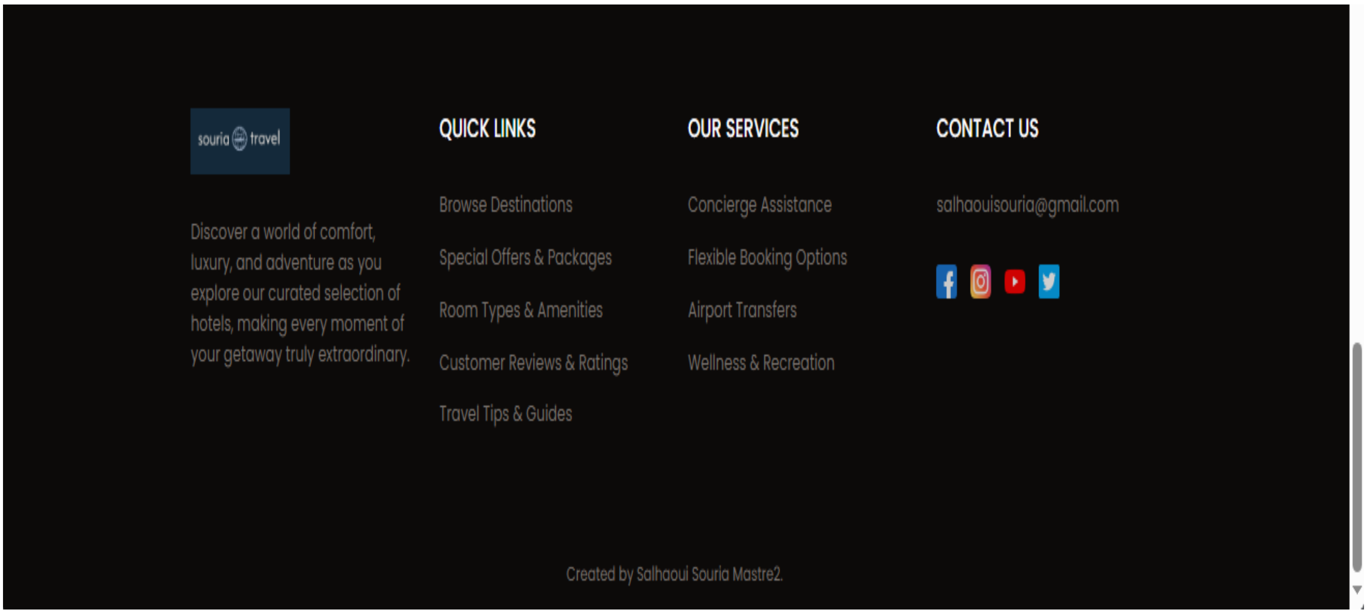


Figure 4.60: Our Contact

4.3.6 Restaurant Services

Figure 4.61 describe our web application Restaurant Services home interface where : In this part, the user tests the state in which he wants to search for restaurants, and by selecting it, a list will automatically appear to him containing all the restaurants that contain it. After that, the user clicks on the restaurant he desires.

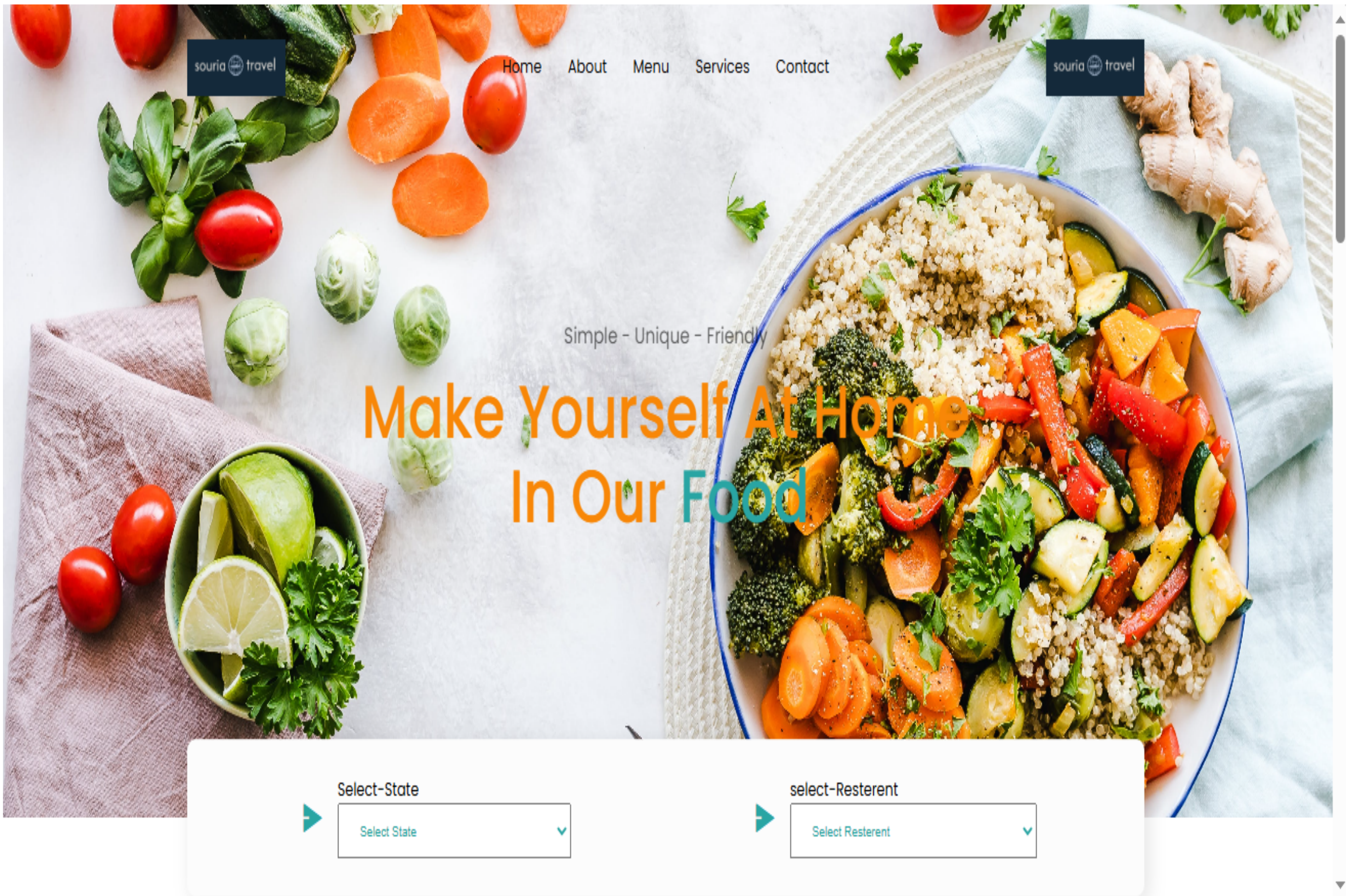


Figure 4.61: Home Restaurant service

1. **About :** Providing a short Description about Restaurant and their information .As shown in picture 4.62

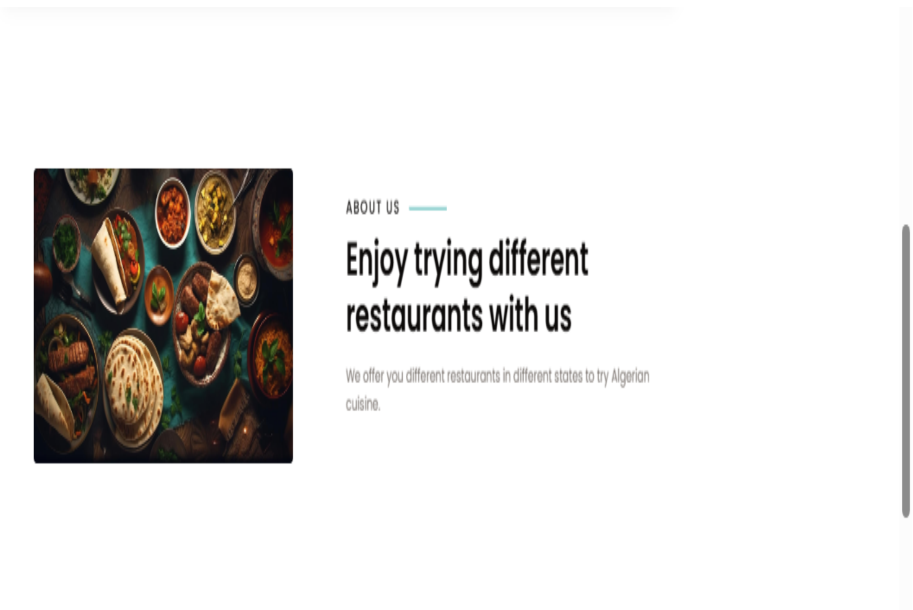


Figure 4.62: About Restaurant service

2. **details :** When the user clicks on the selected Restaurant he wants, the detailed information for the Restaurant appears in this part, which is shown in the following image 4.63

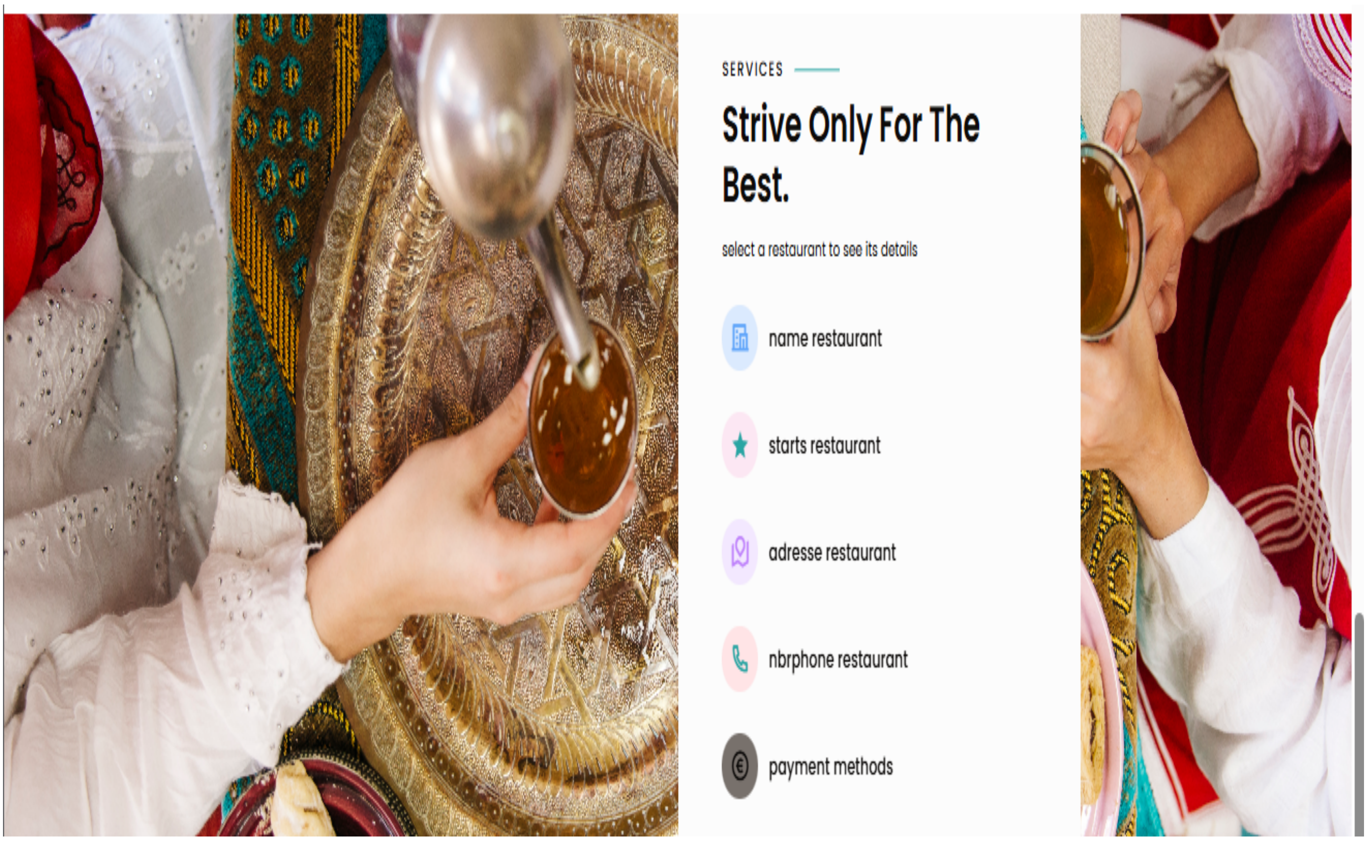


Figure 4.63: details Restaurant service

3. **Contact :**Our information to contact us . As shown in picture 4.64

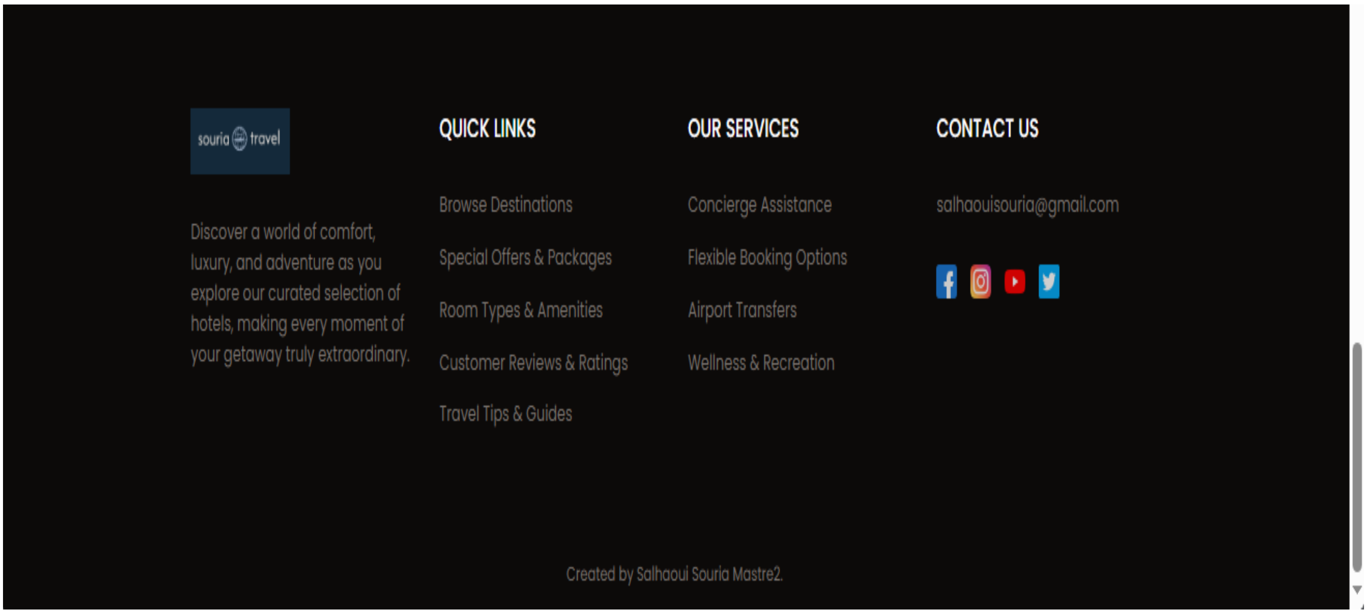


Figure 4.64: Our Contact

4.3.7 places Services

This is a page for the places service. Image 4.65 shows the interface, where if someone wants to know specific information about the places he wants, he will find its details in Image 444, explained in detail.

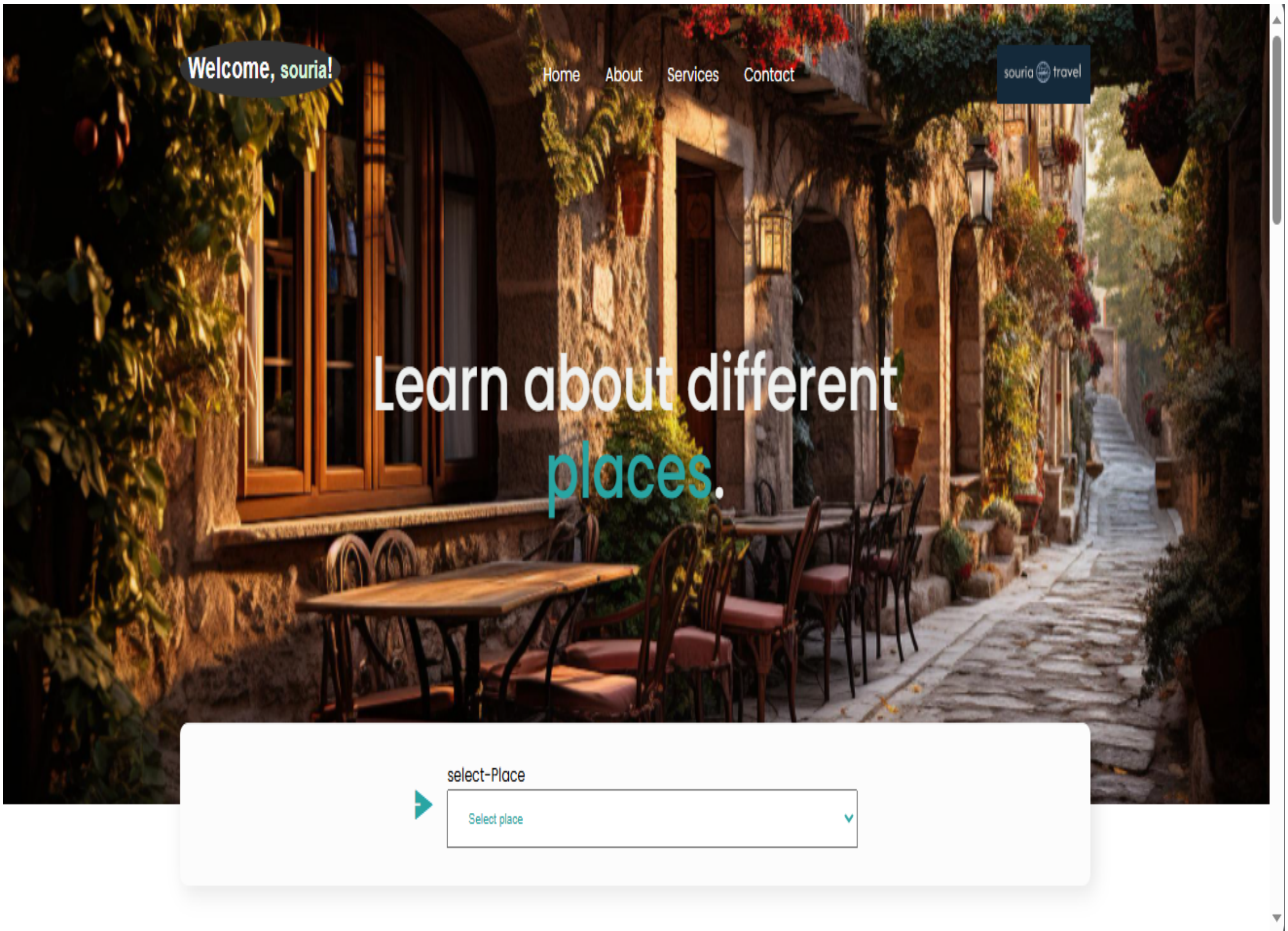


Figure 4.65: places Services page

- **About :** In this part there is an introduction to travel and places .shown as follows in image 4.66

Figure 4.66: About places Services

- **Recommendation service for the best places:** In this part, the results of recommending places to the user are based on his previous comments about the places.shown as follows in image 4.67

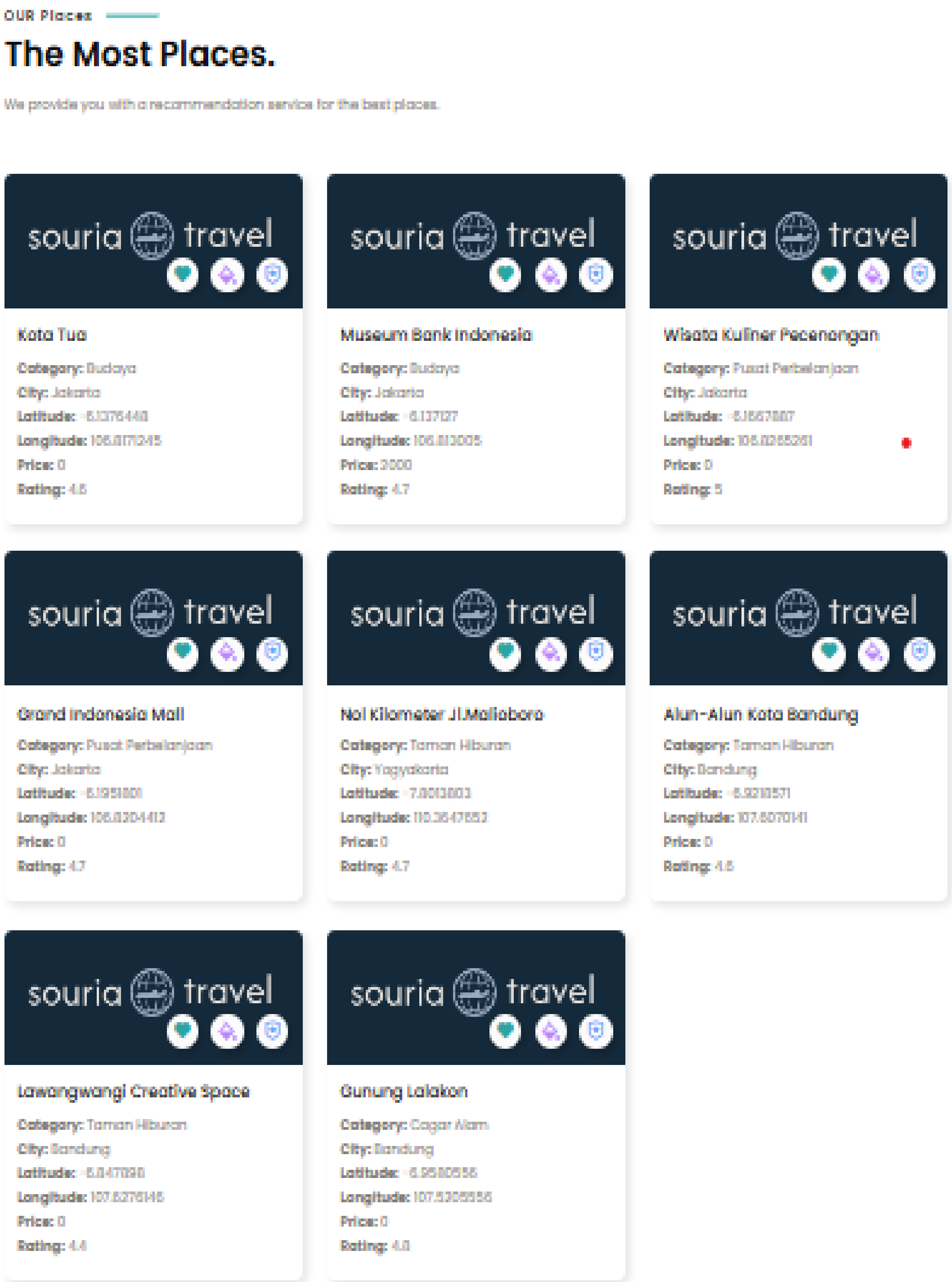


Figure 4.67: Recommendation service for the best places.

- **details :** In this part, there is a card that shows detailed information to the user about the places, shown as follows in image 4.68

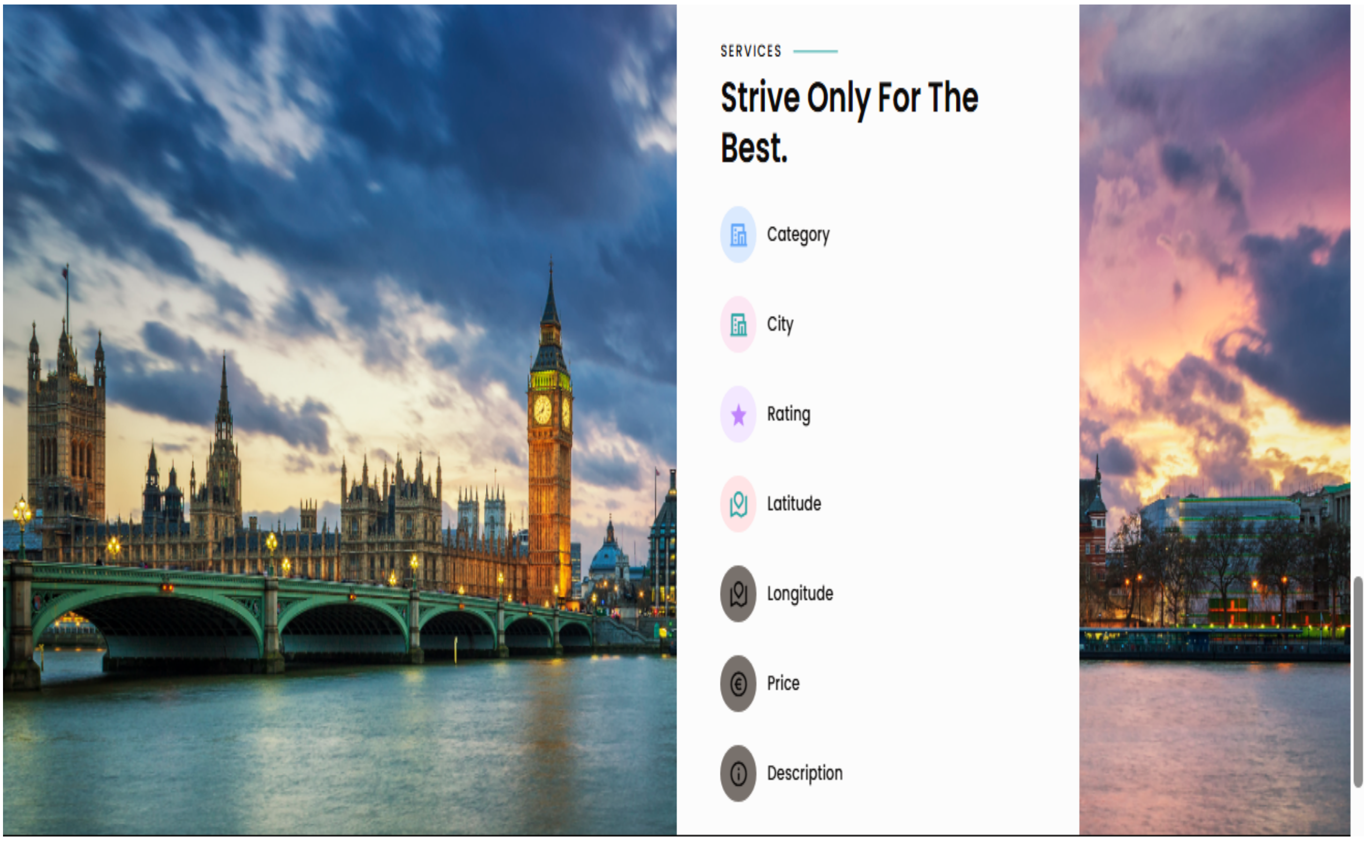


Figure 4.68: details places Services

- **Contact :**Our information to contact us . As shown in picture 4.69

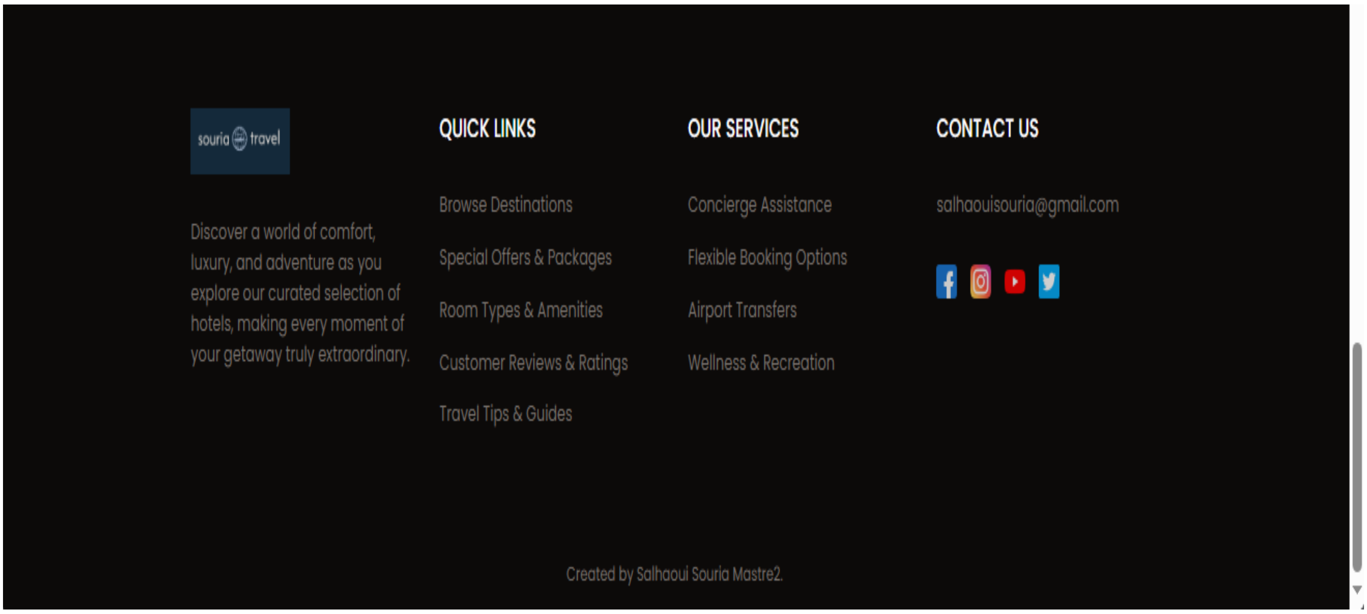


Figure 4.69: Contact places Services

4.3.8 Recommendation score for different users

We see in the following two images the result of a recommendation for users to clarify the difference between them through their previous likes. We took an example of random users.

user souria :

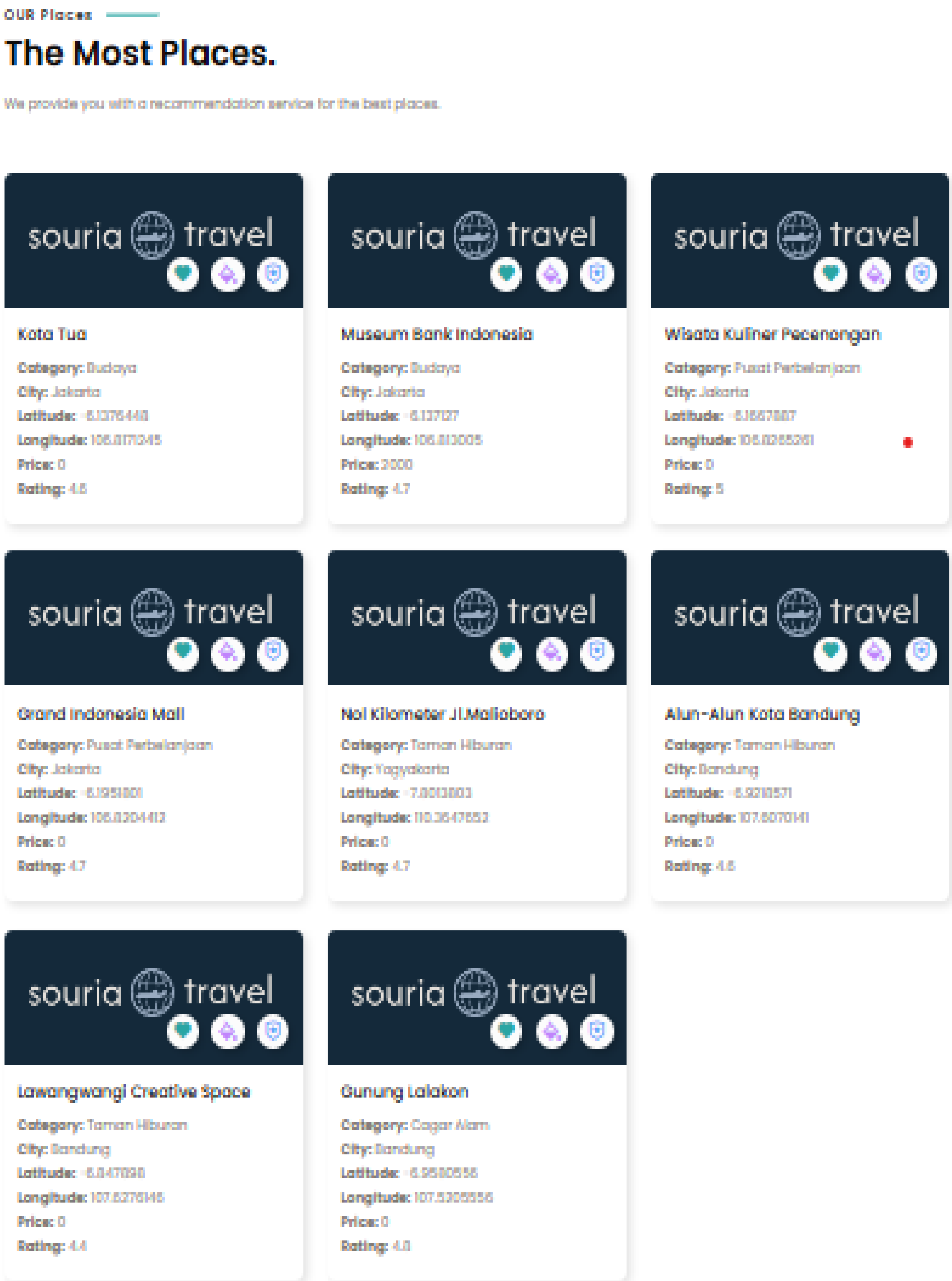


Figure 4.70: User souria recommendation score .

user 60 :

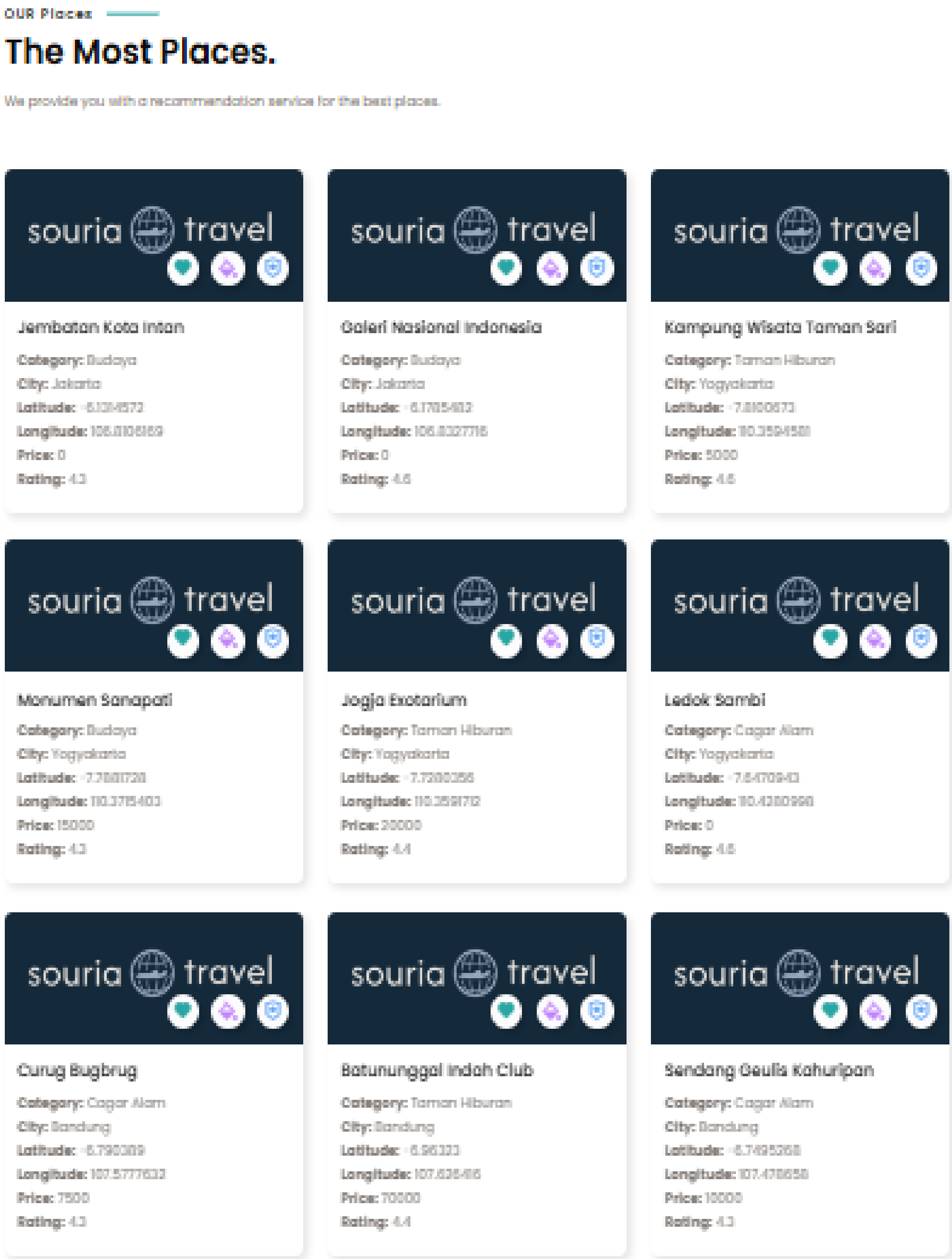


Figure 4.71: User 60 recommendation score .

4.4 conclustion

In this chapter, we first introduce the hardware tools used, then introduce our intelligent model that predicts travel recommendations and finally introduce our main interfaces for our web application and the results of our travel recommendation system.

General conclustion

In the modern travel industry, the ability to understand and cater to individual customer needs has become a cornerstone for success. This project focuses on developing a sophisticated system that harnesses machine learning to enhance the travel planning experience by evaluating customer preferences and budgets. By comprehensively analyzing customer data, the system aims to discern the desires and financial constraints of travelers, thus enabling it to recommend the best tourist destinations tailored to each individual's interests.

The project goes beyond mere destination recommendations by providing tourism offers that closely align with the unique desires of each customer. This personalized approach ensures that travelers receive suggestions that resonate with their specific tastes, enhancing their overall satisfaction and experience. Additionally, the system endeavors to assess and optimize prices, adjusting them in accordance with each customer's budget. This dynamic pricing strategy not only makes travel more accessible but also maximizes the value for each traveler, ensuring a memorable and financially feasible trip. Through these innovations, the project aspires to transform the way people plan and enjoy their travels, offering a seamless, customized, and budget-friendly experience.

Future work for this project includes several key areas of development: integrating additional data sources such as social media trends, real-time travel reviews, and local events to provide more accurate and up-to-date recommendations; developing algorithms that can adapt recommendations in real time based on dynamic factors such as weather conditions, political stability, and emerging tourist destinations; improving the user interface and experience to make the system more user-friendly; including environmental impact assessments to promote eco-friendly travel options; improving the dynamic pricing model using advanced machine learning techniques to provide more competitive and tailored pricing options; and expanding the geographic scope of recommendations to include more diverse and lesser-known destinations. By addressing these areas in future iterations, the travel recommendation system can continue to evolve, providing more accurate, dynamic, and user-centric travel planning solutions.

Bibliography

[1] J. Lu, Q. Zhang, and G. Zhang, *Recommender System Concepts*. World Scientific, 08 2020, ch. 1, pp. 3–14. [Online]. Available: https://www.worldscientific.com/doi/abs/10.1142/9789811224638_0001

[2] H. Ko, S. Lee, Y. Park, and A. Choi, "A survey of recommendation systems: Recommendation models, techniques, and application fields," *Electronics*, vol. 11, no. 1, 2022. [Online]. Available: https://www.mdpi.com/2079-9292/11/1/141

[3] *Overview of Machine Learning and Financial Applications*, ch. Chapter 1, pp. 1–13. [Online]. Available: https://www.worldscientific.com/doi/abs/10.1142/9781786349378_0001

[4] P. Resnick and H. R. Varian, "Recommender systems," *Communications of the ACM*, vol. 40, no. 3, pp. 56–58, 1997.

[5] C. Janiesch, P. Zschech, and K. Heinrich, "Machine learning and deep learning," *Electronic Markets*, vol. 31, no. 3, pp. 685–695, 2021.

[6] D. P. . C. Hashemi-Pour, "Ai vs| machine learning vs| deep learning: Key differences," 2023. [Online]. Available: https://www.techtarget.com/searchenterpriseai/tip/AI-vs-machine-learning-vs-deep-learning-Key-differences

[7] J. Ramos, "The impact of machine learning on travel recommendations," 2022.

[8] C. Aggarwal, *Recommender Systems: The Textbook*. Springer International Publishing, 2016. [Online]. Available: https://books.google.dz/books?id=GKjWCwAAQBAJ

[9] F. Isinkaye, Y. Folajimi, and B. Ojokoh, "Recommendation systems: Principles, methods and evaluation," *Egyptian Informatics Journal*, vol. 16, no. 3, pp. 261–273, 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1110866515000341

[10] N. Kumar, K. V. Pallavi, and B. R. Hanji, "Personalized travel recommendation system using average cumulative rating matrix factorization technique: Concept and framework," *Vietnam Journal of Computer Science*, vol. 10, no. 02, pp. 159–195, 2023. [Online]. Available: https://doi.org/10.1142/S2196888822500361

[11] A. Da'u and N. Salim, "Recommendation system based on deep learning methods: a systematic review and new directions," *Artificial Intelligence Review*, vol. 53, no. 4, pp. 2709–2748, 2020.

[12] *Introduction to Machine Learning*, ch. Chapter 1, pp. 1–22. [Online]. Available: https://www.worldscientific.com/doi/abs/10.1142/9789811201967_0001

[13] J. H. Fetzer, *What is Artificial Intelligence?* Dordrecht: Springer Netherlands, 1990, pp. 3–27. [Online]. Available: https://doi.org/10.1007/978-94-009-1900-6_1

[14] I. El Naqa and M. J. Murphy, *What Is Machine Learning?* Cham: Springer International Publishing, 2015, pp. 3–11. [Online]. Available: https://doi.org/10.1007/978-3-319-18305-3_1

[15] C. W. By Patrick Grieve, "Deep learning vs. machine learning," November 14, 2023. [Online]. Available: https://www.zendesk.com/blog/machine-learning-and-deep-learning/

[16] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[17] J. D. Kelleher, *Deep learning*. MIT press, 2019.

[18] L. Deng, D. Yu *et al.*, "Deep learning: methods and applications," *Foundations and trends® in signal processing*, vol. 7, no. 3–4, pp. 197–387, 2014.

[19] "Python correlation matrix. retrieved from," (Accessed on May 16, 2023). [Online]. Available: https://datagy.io/python-correlation-matrix/

[20] M. K. Lyst, "Metadata embeddings for user and item cold-start recommendations," 30 Jul 2015. [Online]. Available: https://arxiv.org/pdf/1507.08439

[21] an 21, 2023. [Online]. Available: https://scholar.google.com/scholar?q=feedback+in+recommender+system&hl=en&as_sdt=0&as_vis=1&oi=scholart

[22] Feb 18, 2021. [Online]. Available: https://medium.com/analytics-vidhya/recommender-systems-explicit-feedback-implicit-feedback-and-hybrid-feedback-ddd1b2cdb3b#:~:text=One%20type%20of%20feedback%20is,after%20observing%20the%20users'%20behavior.

[23] "Welcome to lightfm's documentation!" [Online]. Available: https://making.lyst.com/lightfm/docs/home.html#articles-and-tutorials-on-using-lightfm

[24] "Lightfm: A hybrid recommender system and its business applications," Sep 30, 2022. [Online]. Available: https://mercurylabs.io/p/what-is-lightfm-and-why-its-useful

[25] "Lightfm's documentation!" [Online]. Available: https://making.lyst.com/lightfm/docs/cross_validation.html

[26] J. Casabona, *HTML and CSS: Visual QuickStart Guide*. Peachpit Press, 2020.

[27] I. Andriyanova, D. Maurice, and J.-P. Tillich, "New constructions of css codes obtained by moving to higher alphabets," *arXiv preprint arXiv:1202.3338*, 2012.

[28] W. H. Mitchell, "Php," 2008.

[29] A. Friends, "Xampp apache+ mariadb+ php+ perl," *Apache Friends*, 2017.

[30] D. Rolon-Mérette, M. Ross, T. Rolon-Mérette, and K. Church, "Introduction to anaconda and python: Installation and setup," *Quant. Methods Psychol*, vol. 16, no. 5, pp. S3–S11, 2016.

[31] D. Flanagan and P. Matilainen, *JavaScript*. Anaya Multimedia, 2007.

[32] P. Naik, G. Naik, and M. Patil, "Conceptualizing python in google colab," *India: Shashwat Publication*, 2022.

[33] “kaggle.” [Online]. Available: https://www.kaggle.com/discussions/general/328265