Hotel Recommender full stack website

Team Detail

Mentor:- Sudeshna Ghosh

Members:-

1. Jai Satyam Thakur (2162026)
2. Pratham R Sharma (2162034)
3. Rishi Ranjan Sinha (2162024)
4. Dev Vatsa (2162029)

Problem Statement

•Hotel selection can be a time-consuming and overwhelming task for travelers.

•Personalized recommendations based on individual preferences can significantly enhance the user experience.

•Machine learning techniques can help predict and suggest the most suitable hotels based on user interactions, preferences, and hotel attributes.

•This project focuses on developing a hotel recommendation website that uses advanced machine learning algorithms and enhanced web development tools to provide tailored suggestions.

•The aim is to create an efficient and accurate website that improves the hotel selection process and enhances customer satisfaction through personalized, data-driven recommendations.

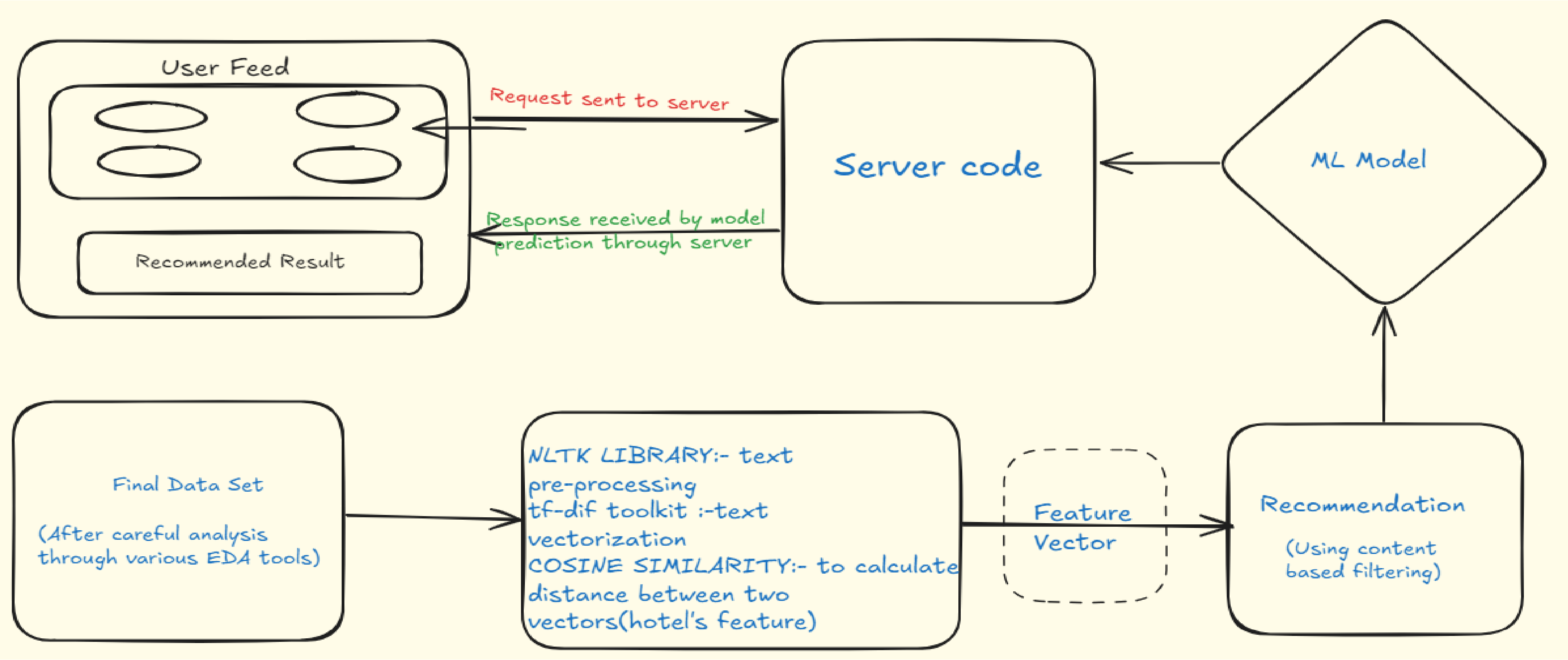
Dataset Overview

The dataset is taken from Kaggle and then pre-processed and has 47693 entries.

Important features are:-

- Hotel ID and Name: Unique identifiers for each hotel.
- Description: Hotel amenities and services.
- Location: Geographical details for map-based recommendations.
- Room Types and Amenities: Available services like Wi-Fi, gym, pool, etc.
- Rating: User-generated ratings (1-5 scale).

Proposed Methodology



Data pre-processing

- Missing hotel ratings were filled with the average rating using Pandas to ensure data completeness.
- Records missing critical attributes, such as hotel names, were removed using Pandas to maintain dataset quality.
- Hotel descriptions and amenities were tokenized to break text into meaningful units for analysis.
- Stop-words were removed to eliminate irrelevant words that do not contribute to recommendations.
- All text was converted to lowercase using Pandas to ensure uniformity and consistency in the dataset.

Feature Engineering

- Feature Consolidation: Attributes like amenities and room types were combined into a single tags column using Pandas to provide a holistic representation of each hotel.
- Textual Data Transformation: The tags column was preprocessed and vectorized using TF-IDF from Scikit-learn, converting text into numerical features suitable for machine learning.
- Importance Weighting: TF-IDF assigned higher weights to unique terms, ensuring that relevant words carried more significance in the recommendation process.
- Normalization of Ratings: Hotel ratings, a key numerical feature, were scaled to a consistent range.
- Data Preparation for Modeling: The processed textual and numerical features were integrated into the final dataset, ensuring compatibility with the recommendation model.

Exploratory Data Analysis

- Exploratory Data Analysis (EDA) was conducted to understand dataset patterns, identify key features influencing recommendations, and detect data quality issues.
- Pandas was used for data manipulation and statistical analysis, while Matplotlib and Seaborn were employed to create visualizations like bar charts, histograms, and heatmaps.
- Interactive visualizations, such as hotel rating distributions and amenities relationships, were created to provide deeper insights.
- Key findings included identifying popular amenities like free Wi-Fi and visualizing correlations between numerical features such as ratings and locations.
- EDA also revealed missing values, outliers, and duplicates, which were addressed to ensure data consistency and quality.

Model Training

- A content-based filtering approach was selected, focusing on hotel attributes to tailor recommendations.
- TF-IDF (Term Frequency-Inverse Document Frequency) was used to vectorize hotel descriptions, emphasizing significant terms.
- Cosine Similarity was applied to calculate relevance between user preferences and hotel attributes, ensuring accurate rankings.
- Cosine Similarity is a metric that calculates the cosine of the angle between two vectors in a multi-dimensional space, measuring their similarity.
- The dataset was split into 80% training data for model development and 20% test data for evaluation.
- This approach ensures the model learns from hotel data and effectively predicts recommendations based on user input.

Model validation

•The performance of the recommendation model was validated using key metrics such as Precision, Recall, and F1-Score, which evaluate the relevance and effectiveness of the predictions.

•Precision measured the proportion of recommended hotels that were truly relevant to the user’s preferences.

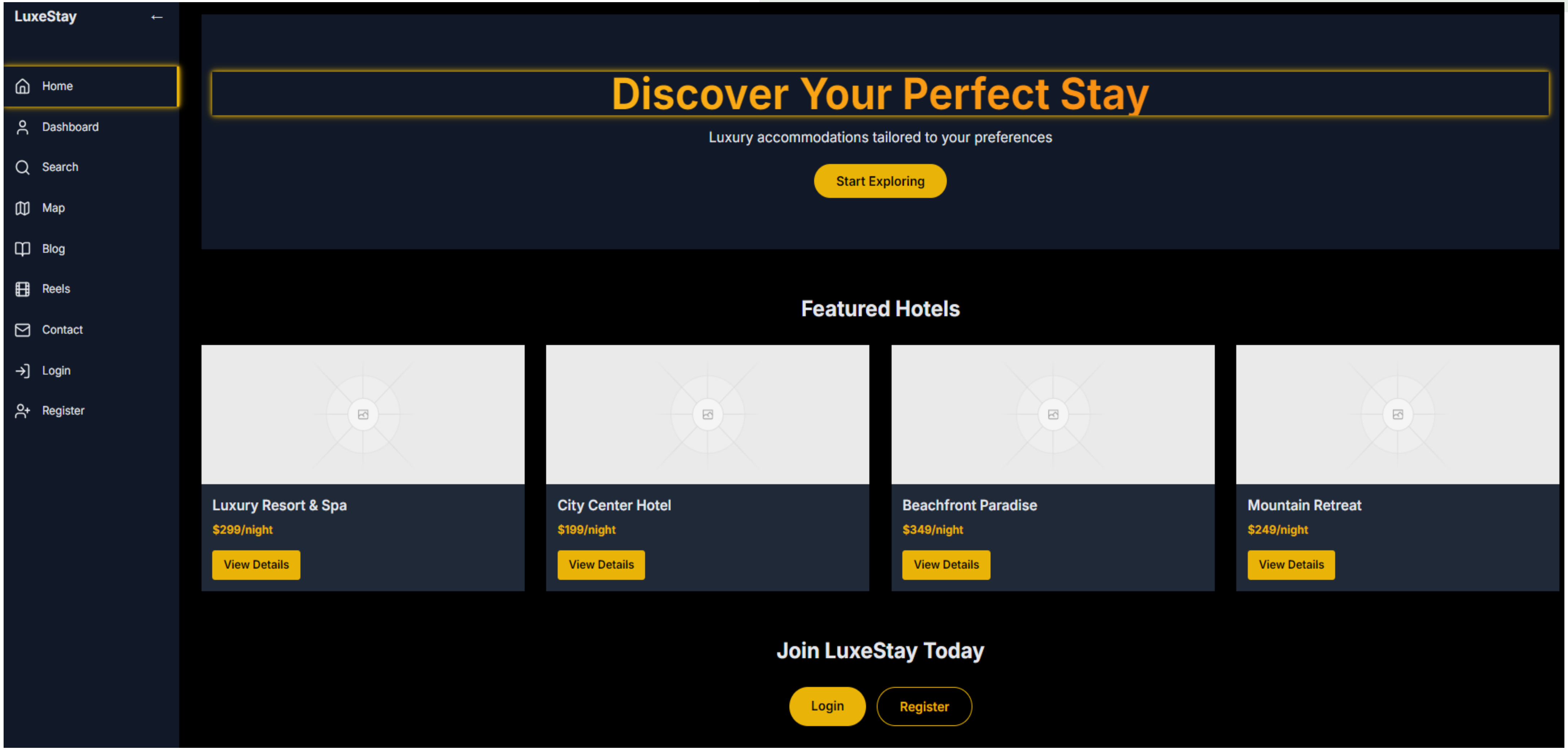
•Recall assessed the ability of the model to identify all relevant hotels from the dataset based on user inputs.

•The F1-Score, as a harmonic mean of Precision and Recall, provided a balanced evaluation of the model's performance.

•Validation results indicated that the model achieved high accuracy, effectively predicting hotels closely aligned with user preferences, enhancing the recommendation quality.

Website development

- Framework: Built using Next.js for server-side rendering (SSR) and static site generation (SSG), ensuring high performance and improved SEO.
- Styling: Implemented Tailwind CSS for a utility-first approach to responsive and visually consistent design.
- Key Features:
- Dashboard: Displays personalized hotel recommendations dynamically.
- Reel Section: Interactive feature for uploading and viewing short videos.
- Map Interface: Enables users to explore hotels based on geolocation.
- State Management: Used React Context API for efficient management of global state across the application.
- User Experience: Designed for responsiveness and intuitive navigation to enhance user engagement.



A snapshot of the website

Recommendation model integration

- The integration process connected the machine learning model to the web application through FastAPI, enabling seamless interaction between the frontend and backend.
- FastAPI served as a bridge, allowing user inputs and preferences to be sent to the backend and processed by the recommendation engine.
- API endpoints were designed to handle requests efficiently, such as fetching saved hotels, processing recommendations, and returning results in real-time.
- The backend dynamically retrieved hotel data, processed it with the recommendation logic, and sent ranked outputs back to the frontend for display.
- This integration ensured a responsive system where user interactions translated directly into accurate, personalized hotel recommendations.

Conclusion

- The project successfully provides personalized hotel recommendations, enhancing the user experience with relevant suggestions.
- Features like the reel section, interactive map, and saved hotels increase user engagement and interactivity.
- The content-based filtering model, integrated with TF-IDF and Cosine Similarity, ensures accurate and efficient recommendations.
- The seamless integration of the frontend, backend, and machine learning model results in a responsive and user-friendly application.
- The system achieves high accuracy in aligning recommendations with user preferences, validated through robust evaluation metrics.

Future work

- Hybrid Recommendations: Combine content-based filtering with collaborative filtering for improved accuracy and diversity in recommendations.
- User Feedback Integration: Implement a feedback loop to refine recommendations dynamically based on user input and behavior.
- Expanded Dataset: Incorporate more hotel data, user interactions, and real-time reviews for broader coverage and richer insights.
- Advanced Features: Add functionalities like booking integration, real-time availability, and price comparison for a complete user experience.
- Scalability Enhancements: Optimize the system for handling a larger user base and datasets without compromising performance.

Thank you