

DSCI 551: Foundations of Data Management

Final Project: Offbeat Travels

Report: Team 74

May 3, 2024



## **Table of contents**

1. Introduction
2. Planned Implementation
  - 2.1. Scalable Data Management System
  - 2.2. Administrative Interface
  - 2.3. Engaging Web Application for Users
  - 2.4. Integration and Testing
3. Architecture Design
  - 3.1. Data Layer
  - 3.2. Application Layer
  - 3.3. Presentation Layer
  - 3.4. Security Features
4. Implementation
  - 4.1. Overview
  - 4.2. Features and Functionalities
  - 4.3. Tech Stack
  - 4.4. Use Case Diagram
  - 4.5. Implementation Screenshots
5. Learning Outcomes
  - 5.1. Full-Stack Web Development Skills
  - 5.2. Practical Application of Theoretical Knowledge
  - 5.3. Software Development Best Practices
  - 5.4. Challenges Faced
6. Individual Contribution
7. Conclusion
8. Future Scope

Link to Drive:

<https://drive.google.com/drive/folders/1xZddcA5xc3tZo4uklnJcdmSLcv3QUORx?usp=drive link>

## 1. Introduction

In today's digital age, the travel industry is experiencing a significant transformation, fueled by the increasing demand for unique and personalized travel experiences. Traditional travel platforms often focus on popular destinations, leaving a niche for explorers interested in the road less traveled. Recognizing this gap, our project, "Offbeat Travels," aims to offer a distinctive solution by connecting travelers with unique, lesser-known destinations through a robust and user-friendly web application.

"Offbeat Travails" leverages advanced technology to handle diverse travel-related data across a scalable, distributed database system. This approach not only ensures efficient data management and retrieval but also supports scalability as the platform grows in popularity and data volume. Our project is built around two main components: a backend management interface for database operations and a frontend web application that provides an engaging user experience for travel discovery and booking. This report details the journey of developing "Offbeat Travels" from conception to implementation. We discuss the challenges faced, the solutions implemented, and the collaborative effort of our team members, Pardi Bedirian and Christopher Bou-Saab, who brought their expertise in backend development, databases, and frontend design to the forefront of this project. Through "Offbeat Travels," we aim to redefine how travelers explore new destinations, making the less traveled roads more accessible and enticing. The following sections will delve into the specifics of our system's architecture, the design process, and the innovative features that set "Offbeat Travels" apart in the competitive landscape of travel technology.

## 2. Planned Implementation

The planned implementation of "Offbeat Travels" was meticulously structured to address both the technological and user experience demands of a modern travel platform. This section outlines the strategies and steps we proposed to develop and integrate the components of our travel discovery and booking system.

### 2.1. Scalable Data Management System

The cornerstone of our project is a scalable data management system that integrates SQL and NoSQL databases to manage a wide range of data types and structures. The SQL component handles structured data such as user profiles, bookings, and flight details, while the NoSQL component manages more flexible data like user reviews and FAQs.

### 2.1.1. Database Partitioning

Data is partitioned across multiple databases using hash values to ensure efficient data distribution and retrieval. In our application, efficient data distribution is critical to achieving scalability and performance. To address this, we implement a hashing mechanism to partition users and their associated bookings across multiple databases, namely `offbeat_db1` and `offbeat_db2`. Each user is assigned to a database based on a hash value computed from their username. This hashing is performed using a simple but effective algorithm that maps each username to an integer, which we then use to determine the database assignment by taking the modulo with the number of available databases. This approach ensures a balanced distribution of data, reducing the likelihood of hotspots and enabling more efficient data retrieval and management. By dynamically selecting the database connection at runtime based on this hash, our system can scale horizontally as user demand and data volume grow.

### 2.1.2. Replication and Redundancy

To ensure high availability and reliability, data is replicated across different nodes. This setup minimizes the risk of data loss and improves query response times across the platform.

## 2.2. Administrative Interface

A key feature of our implementation plan is the development of an administrative interface designed specifically for database managers. This interface facilitates the seamless management of data operations, such as insertions, deletions, and modifications.

### 2.2.1. Direct Database Interaction

Administrators can directly interact with the database through a user-friendly dashboard that abstracts complex SQL and NoSQL operations.

### 2.2.2. Real-time Data Monitoring

The interface includes tools for monitoring data health, transaction logs, and performance metrics, ensuring that administrators can quickly identify and resolve issues.

## 2.3. Engaging Web Application for Users

The user-facing component of "Offbeat Travels" is designed to be intuitive and engaging, encouraging users to explore and book travels to off-the-beaten-path destinations.

#### 2.3.1. Dynamic Content Delivery

The application dynamically displays travel options and details, personalized to each user's preferences and past interactions.

#### 2.3.2. Interactive Elements

Users can interact with maps, reviews, and photos, all integrated seamlessly into the platform to enhance the booking experience.

#### 2.3.3. Responsive Design

The web application is optimized for various devices and screen sizes, ensuring a consistent and accessible user experience across desktops, tablets, and smartphones.

### 2.4. Integration and Testing

Throughout the development process, continuous integration and testing are employed to ensure that new features and updates do not disrupt the functionality of the existing system.

#### 2.4.1. Automated Testing

We use automated tests to validate the functionality, performance, and security of both the database systems and the web application.

#### 2.4.2. User Acceptance Testing

Before launch, real users are invited to test the system to gather feedback on usability and to identify any potential issues from a user's perspective.

## 3. Architecture Design

Our application "Offbeat Travels" employs a multi-tier architecture designed to separate the concerns of data management, business logic, and user interface, ensuring high cohesion and low coupling between components.

### 3.1. Data Layer

#### 3.1.1. SQL Database (MySQL)

Hosts structured data including user profiles, bookings, and flight details.

#### 3.1.2. NoSQL Database (MongoDB)

Manages unstructured data such as FAQs and logs.

## **3.2. Application Layer**

### **3.2.1. Flask Application**

Serves as the central business logic processor, handling requests, executing backend logic, and interfacing with databases. The Flask app uses SQLAlchemy for ORM-based interactions with MySQL and PyMongo for MongoDB interactions.

## **3.3. Presentation Layer**

### **3.3.1. Web Interface**

Provides interactive and dynamic web pages rendered through Flask. It facilitates user registration, login, flight searching, booking, and FAQs. User sessions are managed via Flask-Login for authentication purposes.

## **3.4. Security Features**

### **3.4.1. CSRF Protection**

Flask-WTF is used to secure forms against CSRF attacks.

### **3.4.2. Password Hashing**

Werkzeug is used for hashing and checking passwords to enhance security.

## **4. Implementation**

### **4.1. Overview**

The "Offbeat Travels" platform was designed to offer an innovative solution for travelers seeking unique destinations. The implementation involved building a web application that integrates complex functionalities with a user-friendly interface, supported by a robust backend architecture to handle data efficiently across multiple databases.

### **4.2. Features and Functionalities**

#### **4.2.1. User Authentication**

Secure login and registration system to manage user sessions and access control using Flask-Login and Werkzeug for password hashing.

#### **4.2.2. Dynamic Search and Booking**

Users can search for flights, view available options, and book flights. The system handles searches and transactions through a combination of Flask forms and SQLAlchemy queries to MySQL.

#### 4.2.3. FAQ Management

FAQs are dynamically managed through a MongoDB database, allowing for real-time addition and retrieval of frequently asked questions without affecting the relational database operations.

#### 4.2.4. Booking Management

Users can view, update, or cancel their bookings through a user-friendly interface, with real-time database updates.

#### 4.2.5. Security Features

Integrated CSRF protection in forms to prevent cross-site request forgery attacks, ensuring user data security.

### 4.3. Tech Stack

#### 4.3.1. Frontend

HTML, CSS, and JavaScript were used to create an interactive and responsive user interface. Jinja templates were utilized for dynamic content rendering.

#### 4.3.2. Backend

Python with Flask as the web framework, providing a lightweight and efficient system for route management and server-side logic.

#### 4.3.3. Databases

- **MySQL:** Used for storing structured data such as user information and bookings.
- **MongoDB:** Employed for storing unstructured data like FAQs, providing flexibility in document storage and retrieval.

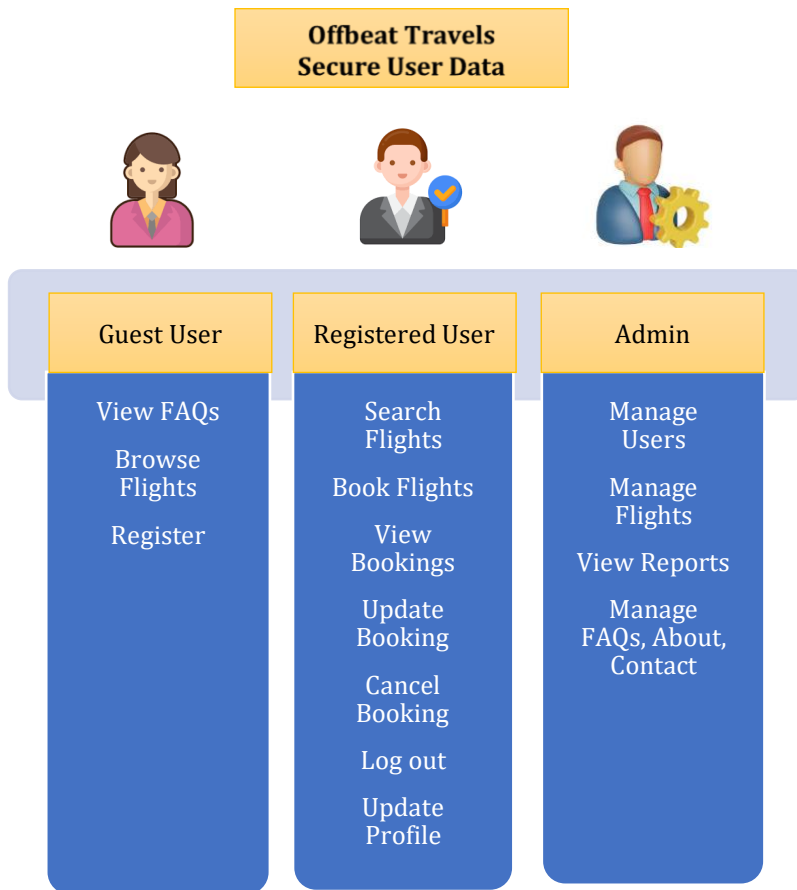
#### 4.3.4. Additional Libraries/Tools

- **SQLAlchemy:** ORM used for database operations on MySQL, simplifying data manipulation and queries.
- **PyMongo:** Used for interacting with MongoDB from Python.
- **Flask-WTF:** For form handling and validations, integrated with CSRF protection.
- **Flask-Login:** For managing user sessions and authentication states.

#### 4.3.5. Development and Deployment

- **Flask Development Server:** Used for local testing and debugging.
- **Logging:** Integrated Python's logging module to track errors and system information.

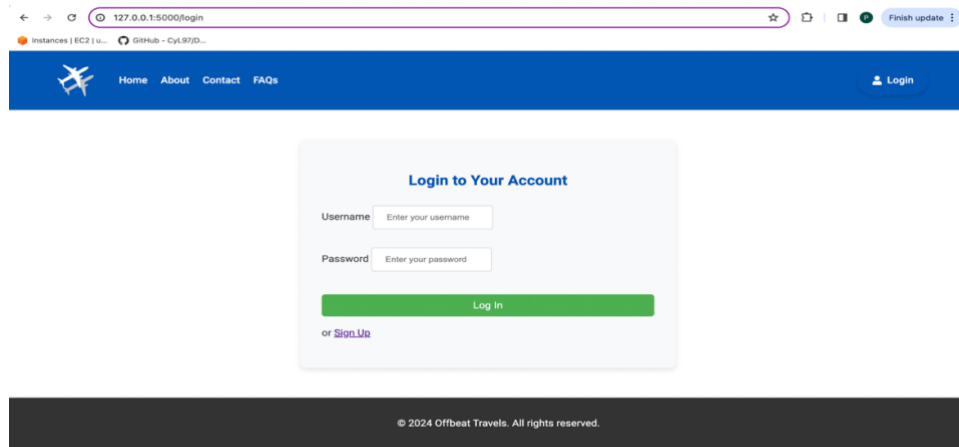
#### 4.4. User Case Diagram





## 4.5. Implementation Screenshots

**Figure 4.5.1 – Login Page:** Showcasing the user authentication form



The screenshot shows a web browser at the URL `127.0.0.1:5000/login`. The page has a blue header with a logo, navigation links (Home, About, Contact, FAQs), and a "Login" button. The main content area features a "Login to Your Account" form with fields for "Username" and "Password", a "Log In" button, and a link to "Sign Up". The footer contains the copyright notice "© 2024 Offbeat Travels. All rights reserved."

127.0.0.1:5000/login

Instances | EC2 | u... GitHub - Cyl97/D...

Home About Contact FAQs Login

**Login to Your Account**

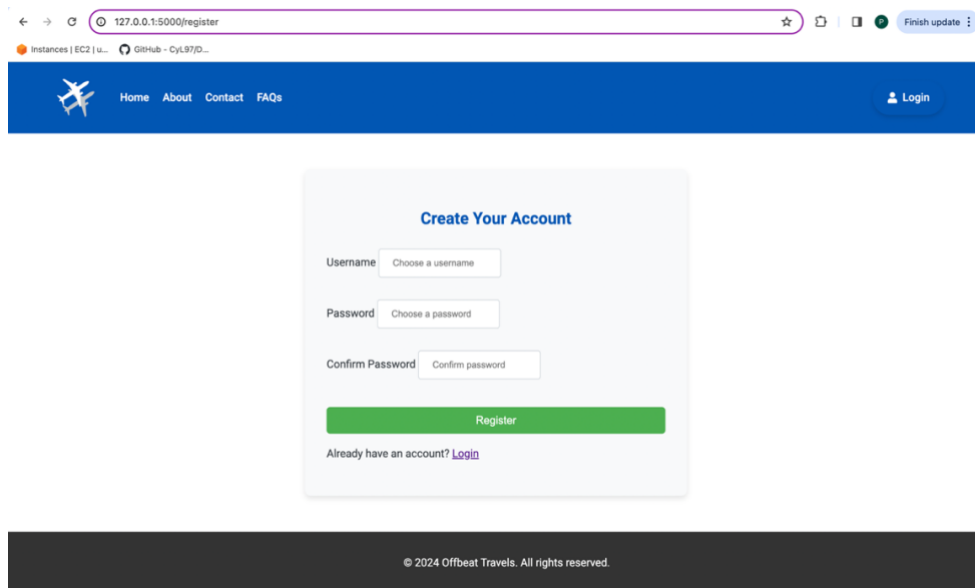
Username

Password

or [Sign Up](#)

© 2024 Offbeat Travels. All rights reserved.

**Figure 4.5.2 – Sign-Up Page:** Displaying the user registration form



The screenshot shows a web browser at the URL `127.0.0.1:5000/register`. The page has a blue header with a logo, navigation links (Home, About, Contact, FAQs), and a "Login" button. The main content area features a "Create Your Account" form with fields for "Username", "Password", and "Confirm Password", a "Register" button, and a link to "Login". The footer contains the copyright notice "© 2024 Offbeat Travels. All rights reserved."

127.0.0.1:5000/register

Instances | EC2 | u... GitHub - Cyl97/D...

Home About Contact FAQs Login

**Create Your Account**

Username

Password

Confirm Password

Already have an account? [Login](#)

© 2024 Offbeat Travels. All rights reserved.

**Figure 4.5.3 – Search Interface:** Displaying the flight search functionalities

127.0.0.1:5000

Instances | EC2 | u... GitHub - Cyl97D...

Finish update

Welcome to Offbeat Travels!

Your gateway to discovering the road less traveled. Explore unique destinations, and immerse yourself in local cultures and experiences.

From: PEK

To: PEK

Departure Date: dd/mm/yyyy

Class: Economy

Search

**Figure 4.5.4 – Booking Confirmation:** A view of the booking confirmation page after a user successfully books a flight

127.0.0.1:5000/my\_bookings

Consulting | Data Science | Jobs | Interviews | Online Courses | DSCI 550 | DSCI 551 | GRDS | DataFirst | Outlook | Gmail | YouTube | Maps | myUSC | All Bookmarks

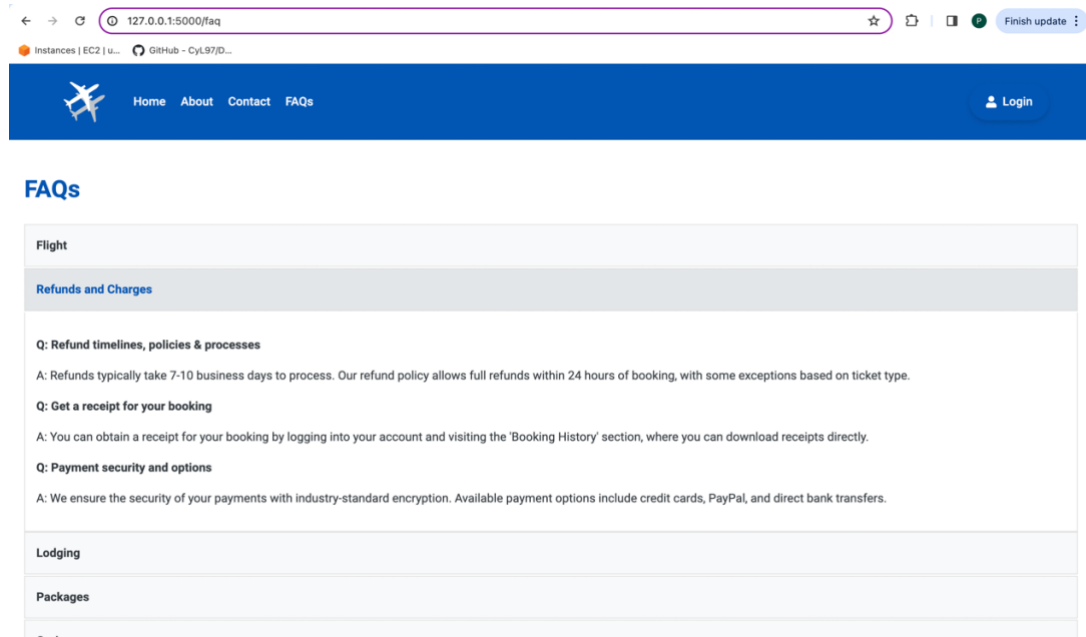
Home | About | Contact | FAQs | My Bookings | Logout

My Bookings

Booking ID	Flight Number	Airline	Origin	Destination	Passenger Name	Family Name	Gender	Date of Birth	Status	Actions
3	CX331	Cathay Pacific	Beijing	Hong Kong	Chris	B	male	2001-04-16	None	<a href="#">Update</a> <a href="#">Cancel</a>

© 2024 Offbeat Travels. All rights reserved.

**Figure 4.5.5 – FAQ Section:** Illustrating how FAQs are presented to the user



## 5. Learning Outcomes

### 5.1. Full-Stack Web Development Skills

#### 5.1.1. Gained Proficiency in Web Development

Building "Offbeat Travels" from scratch allowed us to develop a comprehensive understanding of both frontend and backend development. We learned to effectively use HTML, CSS, and JavaScript to create a user-friendly interface and Flask as a server-side framework to manage business logic, database interactions, and server configuration.

#### 5.1.2. Database Integration and Management

We developed skills in integrating and managing both SQL and NoSQL databases. This included designing schemas, writing efficient queries, and learning how to handle transactions and database security.

### 5.2. Practical Application of Theoretical Knowledge

#### 5.2.1. Data Structures and Algorithms

Implementing search algorithms and data retrieval processes improved our understanding of these concepts in real-world applications.

### 5.2.2. Security Practices

We learned to implement essential web security practices, such as hashing passwords and protecting against CSRF, which reinforced the importance of security in web development.

## 5.3. Software Development Best Practices

### 5.3.1. Vision Control

Using Git allowed us to manage our codebase effectively, understanding the importance of branches and commits for collaborative development.

### 5.3.2. Debugging and Testing

We gained significant experience in debugging code and testing functionalities, which taught us the value of systematic testing in building reliable software.

## 5.4. Challenges Faced

Our project faced several challenges, including the integration of multiple database systems and ensuring the user interface was both intuitive and responsive. In our project, integrating multiple databases—MySQL for structured data and MongoDB for dynamic content—posed a significant challenge. Even within MySQL, we implemented a hash function that would partition user data, including user bookings, across two different MySQL databases. We utilized specific Python libraries for seamless interaction between the databases, ensuring robustness and flexibility. We also faced issues with data consistency and performance as the project scaled. Additionally, we fortified our application with comprehensive security measures like CSRF protection to safeguard user data. This project not only tested our technical skills but also deepened our understanding of complex system implementations, preparing us for future challenges. Despite these obstacles, we achieved a fully functional system that was demonstrated during the in-class demo and received positive feedback for its performance and design. As we conclude this project, we are proud of the technical and collaborative skills gained through this endeavor. The experience has deepened our understanding of distributed databases, web application development, and the practical implementation of complex systems.

## 6. Individual Contribution

### 6.1. Christopher Bou-Saab

Database Design and Structure:

- Architectural Planning: Christopher led the design of the relational database schema for MySQL, ensuring it was capable of efficiently handling structured data such as user information, bookings, and flight details.

- Database Integration: He implemented the integration of both SQL and NoSQL databases into the Flask application, configuring SQLAlchemy for MySQL and PyMongo for MongoDB. This approach enabled effective management of different types of data.

#### Data Collection and Database Population:

- Data Sourcing: Responsible for gathering and vetting data sources to populate the databases, ensuring the data was comprehensive and accurate for the travel offerings.
- Data Insertion and Management: He developed scripts to populate the databases and conducted regular updates and maintenance to keep the data relevant and accurate.

#### Project Structure Organization:

- Codebase Organization: Christopher organized the project structure, ensuring the codebase was well-organized and modular, facilitating easier maintenance and scalability.
- Documentation: He documented the database schemas and provided detailed explanations of the data flows within the application, supporting ongoing development and future troubleshooting.

## 6.2. Pardi Bedirian

#### Frontend Development:

- UI/UX Design: Pardi designed the user interfaces, creating visually appealing and intuitive layouts that enhanced user interaction. She employed HTML, CSS, and JavaScript to implement the designs, focusing on responsiveness and accessibility.
- Interactive Features: She implemented key dynamic features such as the booking system, search functionality, and user authentication flows, which are central to the user experience.

#### Backend Integration:

- Flask Integration: Pardi integrated the frontend with the Flask backend, setting up routes and controllers to efficiently handle user requests, including form submissions, user sessions, and server responses.
- Functionality Testing: She conducted extensive testing of all features to ensure functionality was as expected, debugging, and resolving issues throughout the development and testing phases.

Aesthetic and Functional Enhancements:

- CSS Styling: Pardi was instrumental in styling the application, using advanced CSS techniques to create a cohesive and attractive design theme throughout the platform.
- JavaScript Enhancements: She enhanced the interactivity and responsiveness of the UI with advanced JavaScript implementations.

## 7. Conclusion

In this DSCI 551 project, we successfully designed and implemented "Offbeat Travels," a sophisticated distributed database system that efficiently manages large datasets across both SQL and NoSQL databases. This system not only addresses the challenge of handling vast amounts of data through intelligent partitioning and data management strategies but also provides robust user interfaces for both database managers and end-users.

Throughout the project, we developed two key interfaces to enhance functionality and streamline data management within "Offbeat Travels." The Database Manager Interface is crucial for the administration of the databases, allowing for the insertion, deletion, and modification of data. This interface ensures that data is accurately directed to the appropriate database based on a hash value of the partition key, showcasing the practical application of distributed database principles in a real-world setting.

Additionally, the End-User Web Application provides a dynamic and interactive platform for travelers seeking unique and less-explored destinations. This interface not only boosts user engagement but also exemplifies the seamless integration of backend data management with frontend functionality, creating a cohesive and user-friendly environment for users to plan and book their travels.

Looking forward, we are committed to further refining the application, exploring advanced data partitioning strategies, and enhancing the user experience. This project not only fulfills the academic requirements of DSCI 551 but also provides a solid foundation for future innovations in the field of database management and web development.

This conclusion effectively wraps up the project by summarizing the achievements, reflecting on the learning process, and looking ahead to future improvements, aligning with the academic and practical goals of the course.

## 8. Future Scope

The "Offbeat Travels" project lays a solid foundation for further development and enhancement. As we look ahead, our focus will be on several strategic areas to enhance functionality and user engagement:

#### Enhanced Data Management:

- Implement adaptive data partitioning to improve performance and scalability as user demand grows.
- Integrate real-time data analytics to enable personalized travel recommendations based on user behavior and preferences.

#### User Experience Enhancements:

- Develop a mobile application to cater to on-the-go users, reflecting the increasing use of mobile devices in travel planning.
- Introduce social sharing features that allow users to share their travel experiences and reviews, fostering a community-driven platform.

#### AI and Machine Learning Implementations:

- Utilize machine learning to offer personalized travel suggestions and predictive analytics for demand forecasting, which will help in dynamic pricing and resource allocation.

#### Expansion and Sustainability:

- Promote sustainable tourism options, highlighting eco-friendly travel choices to help reduce the ecological impact of travel.

These enhancements aim to keep "Offbeat Travels" at the forefront of the travel industry, offering cutting-edge solutions that improve both the user experience and operational efficiency while promoting responsible tourism.