UNIVERSITY OF HERTFORDSHIRE

Department of Computer Science
BSc Honours Computer Science
6WCM0029 Computer Science Project

Final Project Report

GoVenture

Acknowledgements

I would like to express my heartfelt gratitude to my supervisor for her invaluable guidance, patience, and support, which played a crucial role in shaping the technical and academic dimensions of this project. I would like to thank her for the endless and continuous iterations of reviewing, developing and going through the report till it was made to the benchmark. I would also like to thank my family for giving me the hope to achieve the goals I had set for myself.

Abstract

This project focuses on the development and evaluation of a cross platform module application designed to simplify flight booking processes while incorporating a smart travel planning feature.

Built using React Native, the application offers core functionalities of user authentication, flight booking and a review section. In addition to this users can generate an itinerary using AI to plan their trips with suggested activities based on the selected destinations.

The development process makes use of a multi step booking workflow to use AsyncStorage and UI components to prioritise accessibility. Manual testing conducted to ensure the user flows to ensure an intuitive and responsive experience and RAGAS was used to ensure that the AI response was precise and relevant.

The findings revealed that the application met many core objectives, however, limitations were identified around the lack of real time features, reducing the overall personalized touch of the application. These results highlighted areas of improvement, to make use of real time APIs and payment gateways for further improvements.

This report provides valuable insights into the challenges of building an end to end travel booking application, offering practical takeaways for a travel tech and AI assisted planning application.

Table of Contents

1. Introduction

In today's fast paced world, efficiency and convenience are essential in travel planning. GoVenture is an innovative online booking application designed to simplify the process of booking flights. Initially, the application was envisioned as a platform where spontaneous flights could be booked but it has evolved into a full fledged booking application where users are shown the available flights, booked flights and reviews of past travel experiences.

1.1. Project Aim

The aim of GoVenture is to develop a user-friendly application that allows users to search for flights, filter them based on visa requirements, book tickets seamlessly, and provide feedback on their travel experience. The application allows the users to make use of an AI itinerary generator for users to plan their vacation.

1.2. Objectives

1.2.1. Core Objectives

- **To design and implement a responsive mobile application**

Offering an intuitive experience across Android and iOS devices while making use of simple minimalistic screens to allow users to be able to navigate through the entire application to reach core features like booking flights and writing reviews with minimal friction.

- **Develop a comprehensive flight browsing and booking system**

Allowing users to explore a list of available flights while ensuring that the page is not too cluttered. USers will be shown the flight names and other details as a scrollable view, clicking in the flight card allows users to see the remaining information about the flight while also allowing the booking of the flight. Other accessibility enhancers allow the filter for 'Visa' and 'No Visa' which enhance the user retention by allowing each card to be filtered, supporting quick decision making and action.

- **To integrate user authentication**

Enabling user account creation and login workflows while ensuring access to personal data. Bookings are saved through the use of the encrypted AsyncStorage, allowing users to be shown their personal activities on the application.

- **To implement persistent booking data and post flight review capabilities**

Users can access a list of booked flights on the homescreen, regardless of the application restating. Once the flight is booked the review should only be possible after the user vacation ends, to reduce any trace of false reviews. This feature adds a post booking engagement layer where vacation itineraries can be generated for better user personalization.

- **Integration of GPT based itinerary generation model**

The AI generated itinerary allows users to transform the application from a simple booking app to an all in one application to holistically help users visualize their trip and prepare better. This objective helps to reflect modern expectations with AI integration in travel and tourism.

- **To enable in app seat selection and ticket customization**

This offers users the ability to book and select their preferred seats accordingly using the modal of the application. This functionality helps improve user experience by simulating real world interfaces to complete the entire digital booking process.

1.2.2. Advanced Objectives

- **To enhance the itinerary generation with tailored recommendations**

Allowing for more meaningful and localization suggestions ensure higher engagement of the application with a touch of personalization. The recommendations made will be referred to the country to be travelling to, with the intention to plan activities around famous and well known locations at the start and then move towards unpopular locations.

- **Explore tone controlled itinerary outputs**

The GPT generated content to be relevant to any person travelling and for any reason, the prompt engineering should involve focussing on the prompt provided and the response generated, to ensure that the tone is relevant to the location and not the user.

- **Maintain booking history**

Reminders or itinerary updates based on the flight date and time to allow a smart and dynamic booking management system that adapts itself to the user's timeline. This makes the application more personalised and lays the groundwork for future enhancements that involve automated check-ins and boarding pass availability as well as trip countdowns to make the application morph into a friendly travel assistant.

- **To test the feasibility of extending GoVenture with future advancements**

The use of AI in the application needs to be integrated and tested to ensure that the responses do not cross boundaries of ethical or legal codes of conduct. The AI should be tailored to analyze the comments of the user’s past reviews and to use them as a base to provide an itinerary of places the user would enjoy more. Further to allow a section that would show users flights that are picked out for them.

- **To notify users when it is the best time to book flights**

The most common issue for users is to be able to plan a trip that is within their budget, with the integration of a model that thinks for the user, user will be notified for when to book their tickets to allow them to purchase them at lower costs, helping them to make smarter financial decisions.

1.3. Summary of Chapters

The report begins with the **Acknowledgement** to supervisor.

The **Abstract** provides a concise overview of the application, its goals and the limitations. The introduction mentions the aims of the application.

Chapter 1: The **Introduction** sets the scene by explaining the context of the project as well as the problem it aims to address. The **Project Aim** and **Objectives** break the project down into the core and advances categories of features it will provide. The **Use of Artificial Intelligence** highlights how AI was integrated into the project.

Chapter 2: The **Literature Review** explores existing research of similar applications to draw comparisons, identify gaps and incorporate the learnings into the project while also examining the current trends and best practices.

Chapter 3: The **Requirements** chapter provides details of the applications functional and nonfunctional requirements to serve as the building blocks for the entire architecture of the application. The **Constraints** chapter discusses the responsibilities and constraints upheld by the project in terms of the ACM and BCS codes of conduct in legal, ethical and professional terms.

Chapter 4: The **Design** chapter presents the structural and visual planning of the application in a way to provide visually pleasing interfaces.

Chapter 5: Followed by the **Technical Development** chapter which offers a breakdown of the code snippets, libraries and tools used in the application.

Chapter 6: The **Evaluation** chapter explains how the testing of the application was executed as well as the automated analysis of the RAGAS tool and Postman for the API.

Chapter 7: The **Results** section presents the raw findings of these evaluations while the **Discussion** section interprets the results while also critically analysing the entire project, mentioning grey areas as well as future improvements.

Chapter 9: The report concludes with a **Conclusion** chapter to reflect on the extent of objectives that were met and those that were not.

Chapter 10: The **Bibliography** and **Appendices** provide supporting documentation of references used within the report.

1.4. Glossary

- **AI (Artificial Intelligence):** This refers to the use of machine learning models to generate the travel itinerary.
- **AsyncStorage:** A local storage used in React Native to securely store the user authentication token.
- **Authentication:** The process of verifying a user's identity.
- **FlatList:** A core React Native component used to render large lists of flight cards displayed on the screens.
- **Filtering:** Displayed on the homescreen to narrow the list of flights that require visas and don't require visas.

- **Flight API:** A third party interface to provide real time flight information.
- **Itinerary:** A travel plan generated by AI to suggest landmarks to visit.
- **Push Notifications:** Messages shown on application in real time.
- **React Native:** The framework used to develop the application for cross platforms.
- **User Interface (UI):** The visual layout of the application.
- **User Experience (UX):** The overall experience users have when using the application.

1.5. Statement about need for Ethics

The testing and evaluation of the project was done by myself. This project does not involve any other human participants, nor any such research or surveys were carried out, therefore according to the guidelines, it does not require ethics approval.

2. Literature Review

In the realm of digital transformation within the tourism industry, (Tutunea, 2019) provides a valuable generational analysis of travelers' preferences in respect to online travel booking systems and tools. Her research was based on 970 Romanian social media users, it notably distinguished the inter-generational user groups who favor adventurous trips using mobiles and rely on social media for travel information. Older generations prefer holiday packages and desktop based softwares that supports the creation of GoVenture which targets mobile first. The study talks in detail about competitor applications like airbnb and Booking.com as the most commonly used services, an identified gap in Booking.com that does not rank it high is the inability to evolve as an application in terms of the UX and UI.

To combat this GoVenture makes use of personalized itinerary makers to give the application a personalized feel. However, the paper's focus was on regional travelers thus it might not reflect the border global trends. Tutunea's research provides a foundational understanding of generational behaviour in online travel booking, serving as a user centred design reference for applications like GoVenture.

Mobile applications have become an integral part to our daily routines, streamlining online shopping and travel bookings, however usability remains a major concern. (Gündüz, 2020) explores usability challenges in mobile flight applications, emphasizing on HCI. The study highlights that while mobile ticketing is available and convenient, users prefer to book flights via PC's. The study goes on in detail about the ways to improve usability through buttons, filters, searches and enhanced visual components of the screen. Similarly (Martin, 2017) introduces the Structuring and Processing Model that acts as a framework to understand the complexities of travel decision making.

Each study focuses on how a user's decision making process is affected by multiple factors The studies together reveal gaps in mobile travel applications particularly in usability and decision making. Gündüz's study identifies usability as a shortcoming that affects recurring customers. The studies add weightage to the workflow of the applications, providing users with clear navigation, options and real time recommendations based on user behaviour. These findings link to my application in terms of the user-friendly interface, allowing users of any age to be able to manage it, making good use of Gündüz's study to make visual clarity of high priority. By bridging these gaps, the application can provide a superior experience for users.

Mobile applications in the sector of tourism have earned significant attention with multiple studies focussing on how technology can support people with disabilities in their tourism activities. In the study of (Ribeiro, 2018), the author explores how mobile computing can go beyond providing location based information to help with tourists disabilities. An overview has been provided in the study of various applications that support accessibility. The study, although comprehensive, also identified the lack of standardized interaction mechanisms and underdeveloped platforms to fully support the tourists.

Another study by (Ismail, 2016) addresses the challenges of tourism in Malaysia. The study commonly pointed out the gaps in current applications that use outdated data, no localization features and lack of GPS integration. To resolve the issues the iTourism app has been

developed to add all the features that were missing from competitor applications both online and offline. The application provides the users with a geotagging system that allows them to navigate their way through their journey. However, the application uses poor UI and does not provide seamless navigation causing it to lag and jitter.

Both the studies focus on providing essential information as well as create an overall experience for users. Both studies provide valuable lessons in terms of the features needed in such applications as well as the critical gaps identified. One key takeaway from these is the necessity of integrating both user centric designs as well as features to make the application more accessible.

The growing need to create cross platform applications has spurred significantly. The study of (Kuitunen, 2019) explores the usage of React Native for building responsive and maintainable applications. The study identifies the major challenge of the need to maintain a separate codebase for different platforms, this increases the development time as well as the cost. React Native bridges this gap allowing users to code using a single code while adding components to different platforms, either iOS or Android.

This mimics my project of coding in React Native, allowing users to use the application over any device with the ease of navigation and not having to worry about the template breaking. The study made it clear that the APK file created is often very large as well as the initial startup time being slower than other applications. Apart from this, there is an issue of the dependencies becoming a liability if they are not actively supported or updated, in such cases the entire application comes to a halt.

Linking these findings to my project, GoVenture an online flight booking application developed in React Native allows any user to experience the smooth navigation transition throughout the application. However, the performance and maintainability concerns could be a threat in the future if the application is further developed into a much more scalable application. The study provides a valuable lens through which to evaluate the application, since it is a small scale application, the rapid development and cross platform reach is essential for user experience. As GoVenture continues to evolve, these considerations will be essential to ensure the efficiency of the application across all the platforms.

The study by Oleksandr (Oleksandr, 2019) explores the evolution of state management in modern web development, emphasizing how it plays a role in scalability and maintainability of application. It highlights the shift for static websites to dynamic, single page applications. The research elaborated on how early web applications managed to function without state tools but as interfaces evolved, state management helped maintain a structured efficient code. It also exerts on the point that if Context APIs are not well used, it can cause issues in the performance of the application thus underlining the necessity of understanding the state tool before using it practically. Go Venture makes use of the React Native state management to toggle between screens of the application, ensuring that each transition to another screen does not affect the experience in any way.

Since the application is in its early stages there is no performance issue but as the application grows the limitations of the Context API can become a concern and for that the study suggests the use of advanced state management tools.

The tourism industry has seen a significant transformation particularly with the growing news to include user central design to attract user engagement and satisfaction. The study by (Putrawan, 2019) explores the implementation of design thinking to promote halal tourism through an application. The design process placed a strong emphasis on collaboration with stakeholders in the cumulation of an application through SystemUsability Scale and User Acceptance Testing. In parallel, the study (Kasinathan, 2017) showcases an android based tourist application utilizing mixed reality to use geotagged information on the real world through a phone's camera. Footprint allows users to explore their point of interest by pointing their phones at landmarks, this offers them real time experience of researching about the specific building through Google or Wikipedia.

Although these studies operate in different technological spaces, they collectively underscore the importance of accessibility, user engagement and to design a user centric application.The convergence of the studies reflect the trend in digital tourism, to create a user tailored application, this pushes my project to strive to create a product that enhances usability.

The study written by Yang (Yang, 2019) explores the vulnerabilities associated with third party in-app payment systems. It identifies the increasing integration of such payment methods in mobile applications and says that it serves as a complex interaction between users, merchants and payment providers. The study examines four major third party payment cashiers into the security rules that they have added to protect such interactions. The researchers found that several attacks were stemming from the violation of the rules, further that hundreds of applications violated at least one security rule.

Yang acknowledges that the sample size is sufficient for summarizing common issues, further analysis could uncover further weaknesses. Since GoVenture focusses on a booking flight app, the security and usage of third party integrations ensures that the identification and mitigation strategies must be involved too.

The study by (Ismail, 2022) presents an examination of the growing reliance on cloud computing since it offers scalability and centralized storage. Side by side it presents critical challenges too, from device mobility to inter-cloud communication. The study identifies the lack of standardization and security solutions that can have a negative effect on the resources of mobile devices. Another threat is the interception or manipulation of data during transmission to the cloud. The application GoVenture leverages mobile cloud computing for key functionalities such as personalized itinerary, generated by the GPT API.

The gaps identified in Ismail's work inform the design decisions that need to be taken for a project, through the use of lightweight and effective encryption protocols to ensure the data flowing is secure. Since the application is small scaled and does not make use of personal information of the user, this paper guides for the mitigation techniques to use in the future.

The study conducted by Londhe (Londhe, 2023) highlights the use of artificial intelligence for personalized travel planning and presents an AI driven solution that leverages Chat GPT for the generation. The paper outlines how conversational AI can be coupled with real time web scraping tools. The content based engine can create tailor travel suggestions based on a user preference such as pricing, layovers and many more. The model used in the study produces a score of 0.86 for precision.

Some gaps such as limited personalization offered by traditional travelling applications have been spotted while other tools provide the basic filters that usually fail to understand the users preferences. The use of ChatGPT significantly improved the reliance on suggestions. Since the project makes use of the GPT API model 3.5, the key valuable insights of this study are essential to the building of my application. Londhe did not only demonstrate the effectiveness of AI in revolutionizing travel planning but also provided a clear roadmap for the incorporation of such features in the GoVenture application, filling critical gaps like real-time responsiveness and personalization. These insights stand to offer a more modern and intelligent companion for the GoVenture application that is beyond just booking flights.

For the development of GoVenture, strong focus is played on UX design principles paired with the Agile project lifecycle to ensure the creation of a user centric and scalable travel platform. UX strategies have been implemented to provide the user with ease in searching and providing feedback. These principles help reduce user frustration and enable a first time user to understand the workflow easily while being responsive across a device matrix. (Ilieva, 2019)

The eXPERT agile methodology, outlined by Ilieva's research, was adopted to implement a small scaled, face paced application designed with changing requirements and high quality product. The requirements of the application included a gap analysis, internal process tailoring and agile training before the pilot implementation was made. Several metrics were inspired from the case study to keep track of the progress and sprint outcomes, maintaining a good balance between the innovation and stability of the application while new features were tested. Through the use of UX design and agile, the development process ensures a continuous alignment of the requirements and the foundation laid by the application to create an application that is based on the user, for the user.

2.1. Use of Artificial Intelligence in GoVenture

The implementation of Artificial Intelligence in the application is to enhance the user experience within GoVenture by providing personalized recommendations and insights. The primary use of AI in this project is through the GPT API, which generates customized travel itineraries for users who have booked a flight. This feature transforms GoVenture from a standard booking application to a personalized travel assistant, helping users make most of the trip by suggesting places to visit and activities to do.

Upon confirming the booking, the application collects details such as destination, travel date and sends them through to the API. The API processes the information and returns a detailed itinerary that is tailored to the user.

3. Requirements

The requirements for GoVenture were defined based on a user-centered approach, ensuring that the application effectively meets the needs of travelers seeking an efficient flight booking experience. To capture and specify requirements a systematic method was followed

1. A user research was conducted understand the needs of modern travellers ad well as that their needs and wants are
2. Existing travel booking applications like Skyscanner and Expedia were examined, identifying the gaps and opportunities for improvement.
3. The development cycles were iterative, following the agile development practices to incorporate new features and changes during the development of the application.

Functional and Nonfunctional requirements were defined to give a holistic view about what is expected from the system and the other aspects of the system, including:

3.1. Functional

- Develop a user friendly interface

Utilizing the React Native language, a cross platform mobile application is developed to work seamlessly for both android and iOS users along with seamless transition between pages throughout the application. The interface prioritizes accessibility and clarity making the process of browsing, navigating and interacting with the application easy.

- User Authentication

GoVenture makes use of a secure encrypted authentication system that allows users to signup and login efficiently. The project does not make use of any database and uses AsyncStorage to store user credentials locally on the device to ensure quick access and persistence of login states. AsyncStorage follows a token based mechanism where credentials are securely stored and are only retrieved when needed. Additionally the entire workflow can only be accessed if authenticated users are logged in, this helps protect the data from being shown to unauthorized users.

- Available Flight List

Currently, the application is dependent on hardcoded airline details and are shown on the homescreen using a FlatList in a card based layout. With the use of flight APIs, the application will show users updated information allowing them to see the real time listing of flights. Key details of the flight will be shown in a collapsed version, by clicking on a card the expanded version of all the flight details are shown.

- Filters

The list of available flights is rendered using a FlatList, which ensures an optimized rendering process, even for a large number of flights. For the ease of users, a visa or no visa filter has been applied to the homescreen where users can select an option to dynamically update the display of flights. The FlightContext is used to manage the state of available flights, it provides seamless access to flight data across different screens without unnecessary re-fetching.

- Flight Booking

The list of flights are listed using a FlatList, which navigates them to a detailed flight screen, the navigation is handled using state management. Once a user books the flight, the data is stored in Asyncstorage, and the state is updated within the FlightContext ensuring the booking status is dynamically updating the application in the app.

- Booking Management

Booked flights are displayed in the Booking Screen, retrieved from the AsyncStorage and managed within the FlightContext. This ensures that even when the application is closed, the data about booked flights persist. The navigation bar is modularized and used among the other screens, this maintains consistency across the different parts of the application. When the card is tapped, the expanded version of the flight details are shown and the option to write a review is shown, with the exception that the reviews can be added once the booking has ended.

- AI Powered Itineraries

After booking a flight, GoVenture integrated the GPT API to generate a personalized travel itinerary. The API request is made asynchronously, to ensure that the UI remains responsive/ The prompt given to the API will be a very specific one to make sure that the response is not biased or focuses on more popular areas rather than others. The AI generated content will show a disclaimer to maintain transparency and users can opt out of receiving AI recommendations if desired. The responses are stored in a state to allow users to revisit their itineraries.

- Review System

Once the flight date has passed, users are prompted to leave a review. The review system ensures that users can review flights after the scheduled travel date, this prevents any premature feedback submissions. Reviews are stored in Asyncstorage ensuring that the application retains user data even after the application is restarted.

- Real time Flight Updates

To prevent users from booking unavailable flights, the application will integrate real time updates or flight availability. This ensures that flight listings are dynamically

refreshed and allows the flight listing to display the number of seats remaining as mitigation technique to prevent any overbooking of the flights.

- Enable Payments

A payment gateway will be implemented for the transactions within the applications. Transactions will be encrypted and follow PCI compliance standards to ensure financial data security.Users will have access to multiple payment methods, including credit/debit cards, e-wallets, and UPI payments. Upon successful payment, a confirmation receipt will be generated and sent via email for record-keeping.

- Push Notification

To keep users engaged and informed, the implementation of real time push notifications can be used to send important updates, from booking confirmations to flight status changes in terms of any delays as well as upcoming flight reminders. Users will be able to experience an application that is customly made for their relevance.

3.2. Non-Functional

- Security & Compliance

GoVenture handles personal user data, which makes it essential to follow the GDPR & CCPA compliance standards. User credentials are encrypted before being stored in asyncStorage, this prevents unauthorized access. This is used since data is only retained as long as necessary.

- Scalability

The GlightContext management ensures that only necessary components re-render when flight data updates, this modular manner allows for future enhancements such as cloud storage or user authentication.

- Performance Optimization

To reduce unnecessary API calls, caching mechanisms should be implemented for flight data and AI generated itineraries. For the time being, the performance is optimized using one API call per day.

- Usability & Accessibility

GoVenture is designed using a user-friendly interface, using React Native’s accessibility features to improve the navigation for users. The entire application makes use of clear and

distinct colours and fonts to help people with disabilities and for them to use the entire application easily.

- AI Ethics

The response of the AI generated itinerary will include a disclaimer to convey to the users that the response is just a suggestion given by the AI model, users will have the option to opt out of the features, this ensures transparency. The GPT response will all be anonymous to ensure user privacy.

- APIs

To enhance the application's functionality and provide a richer user experience, several third party APIs will be integrated to give the application more purpose than just a flight booking application. A Weather API will display the weather forecasts for the flight destinations, currency conversion API will allow users to view ticket prices in different currencies, while also catering to travelling internationally and a tripadvisor API will be implemented for users to be updated and shown the reviews of specific places of the country of visit.

3.3. System Requirements

3.3.1. Software Requirements

The following list is the system requirements for the project:

- Development: Expo dev
- Language: React Native
- Model: gpt-4o-mini
- API: OpenAI
- Performance Testing: Postman
- API Testing: RAGAS
- Testing IDE: Google Colab

3.3.2. Hardware Requirements

- Processor: Intel Core i5
- RAM: Minimum 8GB
- STORAGE: SSD with at least 256 GB
- OS: Windows 10 or above

Android:

- OS Version: Android 9
- RAM: Minimum 2GB

iOS:

- OS Version: iOS 13
- Device: iPhone 13
- RAM: Minimum 2GB

3.4. Project Constraints

The development and deployment of GoVenture are subject to multiple constraints and requirements, particularly in legal, ethical, professional and environmental domains. To ensure the project adheres to the best practices the ACM and BCS codes of conduct have been aligned to the project.

3.4.1. Legal Constraints

- Data Protection

GOVenture involves user authentication and handling of sensitive user data, it must comply with the data protection laws, especially ACM 1.6 Respect Privacy and BCS 2.2 Respect Privacy Rights. The application collects and processes sensitive user information thus data must be encrypted while being entered and at transmission too. In context to this, users should be informed about the data collection and give them the option to opt out or request the deletion of their data. This feature allows users to have the upper hand since they can permanently delete their accounts and personal data.

- AI Transparency

The GPT API is used to generate personalized travel itineraries based on the booking of the flight. Keeping this in mind, AI generated recommendations must be transparent and not manually curated. To maintain ethical use of AI, the ACM 3.1 Ensure Fairness and BCS 2.3 Maintain User Trust codes should be followed by including disclaimers stating that the suggestions made by the itinerary maker may require manual verification for accuracy. Additionally the user should tap on a CTA to start producing the itinerary list, this ensures that users with consent have given the right for the AI model to curate a list. Additionally, the AI interaction should be done anonymously to improve the service quality without compromising the users personal data.

- Third Party API Compliance

GoVenture makes use of third party APIs which have to comply with the ACM 2.5 Give Proper Credit and BCS 1.2 - Comply with Legislation, each have thrift own terms of use, the GPT API has a usage restriction and cannot be exploited for commercial use beyond permitted limits, to combat this issue, the users will only be given one query a day to

produce the itinerary list, this prevents the overuse of an API while also retaining user attention to the application.

3.4.2. Ethical Constraints

- AI Biasness & Discrimination

Since GoVenture provided AI generated travel recommendations, it must ensure that the response it receives is not in any way biased towards a certain airline or destination. Most AI models can exhibit such behaviour, causing them to exclude less popular destinations while promoting others.The ACM 1.4 Be Fair and Take Action Against Discrimination and BSC 2.4 Avoid Bias are rules that highly focus on the responses of models. To combat this issue, the prompt provided to the AI model will be specific and will mention to focus on the entire location rather than just the popular destinations. Apart from this a disclaimer message will be written telling the users not to blindly follow the response since models can be misleading.

- Responsible Use of AI

The ACM 3.5 Respect Intellectual Property and BCS 1.4 Respect Intellectual Property Rights focus on the prevention of any outdated or incorrect information, since users are dependent on the information provided to them by the application. To prevent this, the application will explicitly warn users that AI generated suggestions are for guidance only and should be cross checked with official sources. Additionally AI will be prompted not to pull data from unauthorized travel websites so as to not lead to intellectual property violations.

- Avoid Dark UX Patterns

The ACM 1.2 Avoid Harm and BCS 2.6 Do Not Mislead Users is an important rule for many travel platforms that use manipulative techniques to create fake urgency notifications. GoVenture will avoid such techniques and maintain user trust and ethical integrity. The details relating to the flight will be clearly displayed and not manipulated to pressure the users.

3.4.3. Professional Constraints

- Software Security Standards

User authentication is implemented in GoVenture, it is necessary to implement robust encryption methods to prevent data breaches or phishing attacks. To prevent this, security audits should be performed to identify vulnerabilities in APIs as well as in the entire platform. The platform should undergo penetration testing to be resilient against cyber threats. Adhering to the rules of conduct ACM 2.9 Design and Implement Secure Systems and BCS 1.6 Ensure Software Safety, ensures that the entire workflow is secure and that the software is safe to use for users.

- Code Quality

The ACM 2.2 improves public understanding and BCS 3.2 produces maintainable code to ensure the application follows the industry best practices such as modular programming, reusable components and state management solutions. This allows developments to easily understand and modify the codebase. Additionally, through comments the entire code can be understood for developers who work on it in the future. Regular code reviews and testing should be conducted to maintain code quality and seamless application performance.

3.4.4. Environmental Constraints

- Reducing Server Load

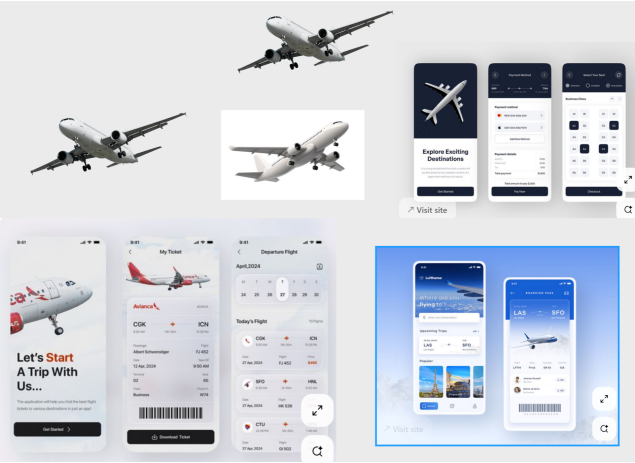
As a cloud based application, GoVenture consumes several resources that contribute to energy consumption and carbon emissions. To mitigate this, the reduced API calls can reduce the strain on the server. Additionally, techniques like lazy loaders and efficient data compression can improve the overall performance of the application, making it eco-friendly to promote sustainability while abiding by the ACM 1.7 and BCS 2.5 rules.

4. Design

The design of the GoVenture application makes use of a simple and minimalistic interface to attract users as well as being easy to navigate. The entire layout is built using React Native to ensure smooth and consistent experiences on any device. The application is structured to have elements strategically placed to guide users step by step from booking to reviewing. The visual hierarchy, colour contrast and text clarity have been carefully considered to make the experience of the application enjoyable.

4.1. Colour Scheme

The colour schemes of the application create both an emotional engagement and guide the user through it. To decide on the colours to use, a mood board was made to find the current booking application structure as seen below:



The current applications make use of shades of blue and thus was incorporated into the application. Once the designs were finalised, the colour scheme was chosen:

GoVenture adopted a distinctive colour palette, the Splash Screen makes use of the vibrant orange colour, giving off a warm and adventurous effect. It sets a friendly and welcoming tone from the start.
In contrast, the navigation bar and buttons make use of the bold and rich navy blue that adds a layer of professionalism, credibility and visual balance. The colour selection enhances user engagement as well as being a great fit for users with partial blindness who are unable to

recognise or read when a colour of different shades is put together, thus targeting diverse groups of users.

4.2. Workflow Design

The workflow of the application has been structured to provide a seamless and intuitive user experience, ensuring that the user does not get confused getting through the app. The initial Login/ Register Screen allows users to either authenticate themselves or create new credentials for the application. A number of error messages on this screen allow users to enter clean data. Once they have entered the correct data, they progress to the HomeScreen that shows the current date being highlighted on the calendar and the filter toggle below it where users can select whether they want to view flights that require a visa or not. The main component of the screen shows the list of the flights in a scrollable card list view, showing the details of the flight. The user can tap on the card to show the full details of the flight. Ensuring that the users can only book a flight once they have read all the details of the flight.

Using simple and interactive components, when the user clicks on the ‘Book’ CTA, they are shown a small modal displaying the number of tickets they want to buy, for both adults and children, by clicking next the user is shown the list of seats, the user can select the seats they would like to book and then they are traversed to a screen for the payment. Since the application is small scaled, the procedure to add the users personal details is skipped for now due to the sensitivity of the data. Once the user books the flight, they are traversed to the Booking Screen.

Using the same UI as the home screen, the booked flights are shown in the form of a card. With two CTAs: one to write a review and the other to generate the itinerary. Users will only be able to write a review when the flight arrival date has passed and the AI itinerary is reduced to one due to the API constraints.

Reviews written are added to the Review screen using the same visuals as used throughout the application

4.3. Interface Design

The design development process went through multiple stages of fidelity, from rough sketches to highly polished screens. The early stages include the wireframes which define the user journey and usability of scenarios before any code was written.

These followed by high-fidelity prototypes which incorporated the basic interactions of the user flows, using the colour, icons and certain images to be used in the application. It is important to note that although the basic flow and UI were followed, the application does not look exactly like the hi-fi designs.

4.4. Design Principles

The design of the GoVenture has been designed using the design principles to form the foundation of the application's user interface and interactive design.

Consistency is maintained throughout the application to help users navigate the app with confidence and ease using colour palettes, consistent typography and standardized buttons. This consistent design approach ensures that users get familiar with the application is structured as well as understanding the applications performance according to the users actions.

The use of modern UX principles ensure that the application is created using simplicity, this eliminates the feeling of a crowded and overused application interface. There are no irrelevant elements being used and the application only shows the essential features that the user can use as well as disabling the buttons when not in use, this helps prioritize functionality over visual clutter.

GoVenture makes use of accessibility very slightly, through the use of two different contrasting colours as well as making sure the buttons on the screen are large, to make touch targets easy for users with limited dexterity. The opposite contrasts of the CTAs across the application ensure that users with a low readability are able to read it, with a navy blue button and white text.

The application gives the user a sense of user control through their choice to choose the flight, ticket number and seat selection. Modals guide the user through a step by step process but never trap them, at any step the user can close the modal.

The application is built on modular components, ensuring a flexible architecture for new addition of features to be made easily. This structure allows future scalability and for the evolution of the application.

5. Technical Development

This section provides a comprehensive overview of the practical implementation and development process of the application, covering the setup aspects, architectural designs, tools, development and overall testing strategies.

5.1. Development Environment

The development was done using React Native due to its cross platform compatibility and reusability component feature using the expo.dev IDE which supports a range of extensions and real time debugging tools.

During the development phase, the IDE allowed for it to be used to view the application using an Android and iOS device using the generated QR code. The choice of React Native aligns with modern trends in application development as highlighted by Smith (2020), who notes its efficiency.

5.2. Version Control

The application did not make any use of Git for version control since the IDE did not allow for me to do it. However, the IDE did make versions of previously saved codes which did achieve the same goal, additionally after adding a new feature, the entire project was downloaded into a zip file, making sure that multiple usable versions of a single project were present.

5.3. Uncertainties in Requirements and Risk Management

- Ticket Limitation

At the outset of the project, several functional and design requirements were ambiguous, especially the seat selection and ticket limitation logic. Both of which have an impact on the experience and technical architecture of the booking flow. In competitor applications, a user can book as many flights as available on a plane, since the application does not make use of an actual real time flight API, the choices for upper and lower bounds were set myself.

To resolve this, iterative prototyping was used to experiment with the user feedback online and going through numerous existing applications that revolved around the same idea. Based on the research, the best practices in airline booking was decided to restrict the bookings to 0-5 for adults and children and to ensure that the input field would not allow the entry of an out of bound value along with disabling the CTA if no passenger seats were selected.

This constraint simplified the seat selection logic that allowed users to select the number of seats with ease

```
<TouchableOpacity
  style={[
    styles.seat,
    isSeatSelected && styles.selectedSeat
  ]}
  onPress={() => {
    if (selectedSeats.includes(seat)) {
      setSelectedSeats(selectedSeats.filter((s) => s !== seat));
    } else {
      if (selectedSeats.length < totalSeatsToSelect) {
        setSelectedSeats([...selectedSeats, seat]);
      }
    }
  }}
  disabled={selectedSeats.length >= totalSeatsToSelect && !isSeatSelected}
>
  <Text style={styles.seatText}>{seat}</Text>
</TouchableOpacity>
</View>
);
})}
</View>
<TouchableOpacity style={styles.nextButton} onPress={() => setStep(3)}>
  <Text style={styles.nextButtonText}>Next</Text>
</TouchableOpacity>
```

- Seat Selection Interface

Another challenge involved the visual layout and structure of the seat selection interface. Since there was no defined specification on how the seats should appear or how many should be made available or whether users would prefer to use visual touchpoints or list based selection, the choice to group it was done by opting for a grid layout. To match the airplane structure, a 6 column grid with 18 total seats was made in groups of threes so as to mimic real world layout as well as create a visual balance on the screen. Additionally, as the user selects a seat, the visual colour changes to the peachy tone.

```
// Create the seat numbers (18 seats in total)
const seatNumbers = Array.from({ length: 18 }, (_, index) => index + 1);
```

```
<View
  key={seat}
  style={[
    styles.seatGroup,
    index % 3 === 0 && { marginLeft: 0 },
  ]}
>
```

Using this code, the seats made use of logic that would adjust the seats ensuring real time feedback, which adhered further to user control and feedback.

- State Management

The basis of the application was to ensure consistent state management across different steps of the application, this was to ensure that users could not bypass steps or add incorrect data into the application. To handle this, multiple validation checks have been used throughout the application along with error messages to keep the user in check at every step of the way

Users are only allowed to book a total of 5 tickets for both adult and children:

```
{(parseInt(adults || '0') + parseInt(children || '0')) > 5 && (
  <Text style={styles.errorText}>You can book a maximum of 5 tickets.</Text>
)}
```

The most important screen, Login made use of them too where the user was not able to proceed until the credentials were added:

```
if (!email || !password) {
  setError('Please enter both email and password!');
  return;
}

if (!validateEmail(email)) {
  setError('Invalid email format!');
  return;
}
```

- Platform Difference

There was an uncertainty around platform specific rendering issues, especially when dealing with UI components like input fields and buttons, which would all result in a bad user experience. To mitigate this, the use of the layout components in SafeAreaView allowed proper rendering across devices.

```
return (
  <SafeAreaView style={styles.container}>
    <Text style={styles.heading}>Login</Text>

    <TextInput style={styles.input} placeholder="Email" value={email} onChangeText={setEmail} />
    <TextInput style={styles.input} placeholder="Password" secureTextEntry value={password} onChangeText={setPassword} />

    {error ? <Text style={styles.errorText}>{error}</Text> : null}

    <TouchableOpacity style={styles.button} onPress={handleLogin}>
      <Text style={styles.buttonText}>Login</Text>
    </TouchableOpacity>

    <View style={styles.signupContainer}>
      <Text style={styles.signupText}>Don't have an account?</Text>
      <TouchableOpacity onPress={() => setActiveScreen('RegisterScreen')}>
        <Text style={styles.signupLink}> Sign up</Text>
      </TouchableOpacity>
    </View>
  </SafeAreaView>
```

- Dynamic Content Management

Across the entire application, content is being dynamically updated, ensuring that selected data is reflected correctly across multiple steps and devices could have led to state inconsistencies. Any small mistake could lead to validation failures and incorrect final selections.
To manage this, the entire application makes use of useStates to control data integrity and ensure that the data is rendered without breaking the application.

```
  const [showBookingModal, setShowBookingModal] = useState(false);
const [step, setStep] = useState(1);
const [adults, setAdults] = useState('');
const [children, setChildren] = useState('');
const [selectedSeats, setSelectedSeats] = useState([]);
const [useSavedCard, setUseSavedCard] = useState(false);
```

- Ensuring that a review can only be written once

During the development process, three issues were observed, a user could write a review of a flight more than one time, the review on the review screen would be overwritten and the user could write a review before they even boarded and experienced the flight. Allowing multiple reviews from the same user could easily lead to lack of data consistency and data integrity as well as confusing the user. To mitigate this, a number of conditional statements were added to check whether the current data was more than the departureTime (data & time) of the flight and with the use of AsyncStorage, the user was permitted to allow only one entry of a review. After writing the review, the review CTA was disabled.

```
{bookedFlights.length === 0 ? (
  <Text style={styles.noFlightsText}>You have no booked flights yet.</Text>
) : (
  bookedFlights.map((flight) => {
    const hasFlightEnded = new Date(flight.destinationTime) < new Date();
```

State variables are used to mark the review as submitted when the user presses the submit button

```
const [submittedReviews, setSubmittedReviews] = useState({});
```

- AI Itinerary Generation

Another issue faced during the development phase was the lengthy response of the API generation. It had formatting issues making it difficult for users to read or understand. To mitigate this a number of prompts were used for it to generate a list of landmarks using bullet points.

```
try {
  const response = await openai.chat.completions.create({
model: 'gpt-4o-mini',
messages: [
  {
    role: 'system',
    content: 'You are a travel assistant that generates short, concise lists of places to visit with no additional descriptions or details. Only provide
the names of the places in bullet points, no elaboration or extra information.',
  },
  {
    role: 'user',
    content: `I am going to ${destination}. Please give me a list of places to visit, no descriptions, in bullet points. Only the names of the places,
5-10 locations max.`,
  },
],
});
```

5.4. Use of Software Tools

Several tools were used to support development:

- Expo.dev: For the development of the application
- Postman: To find the response time of the API
- Figma: Wireframing of the designs

5.5. Testing Strategies

- Tested the API response using RAGAS

RAGAS was used to test the faithfulness, answer relevancy and context precision of the response generated. These three metrics were chosen for the following:

- Faithfulness: this checks to see if the models answer reflects the provided source context without fabricating or hallucinating
- Answer Relevancy: to access how well the generated response addresses the user's query
- Context Precision: measure how well the system utilized the provided context to generate the response.

The answer to the code is shown in a screenshot below, the entire notebook code is added to the appendices

```
result = evaluate(dataset, metrics=[faithfulness, answer_relevancy, context_precision])
```

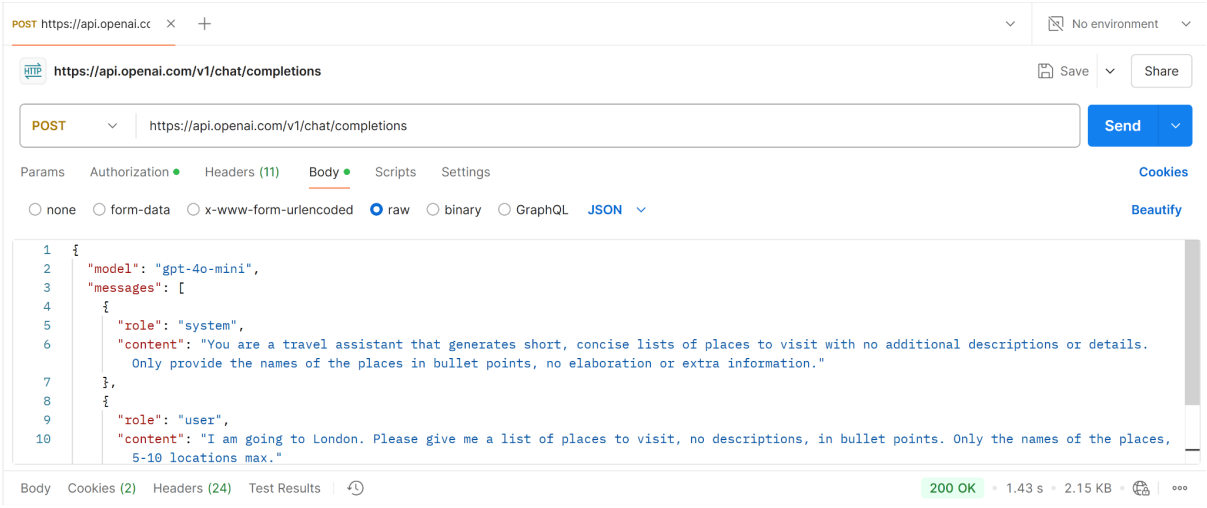
```
Evaluating: 100% 3/3 [00:14<00:00, 4.99s/it]
```

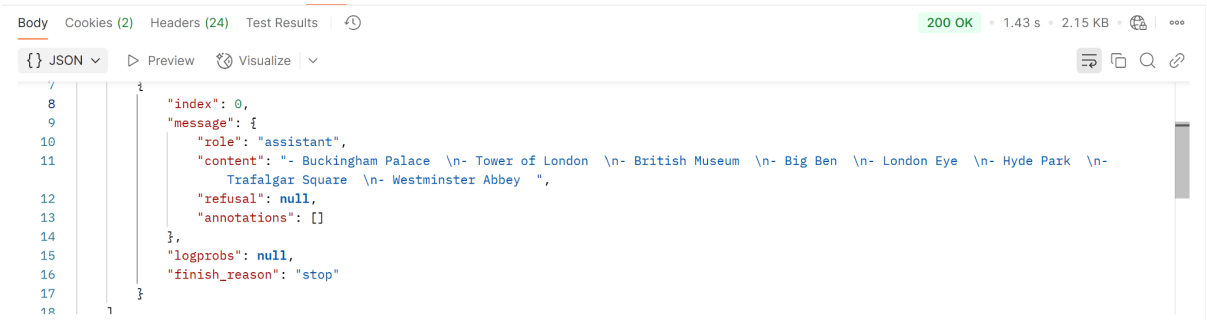
```
print("Evaluation Metrics:")
print(result)
```

```
Evaluation Metrics:
{'faithfulness': 0.5556, 'answer_relevancy': 0.8746, 'context_precision': 0.4405}
```

- Using Postman to check the API response

The API response time and response was tested using Postman

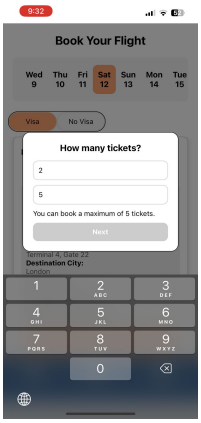
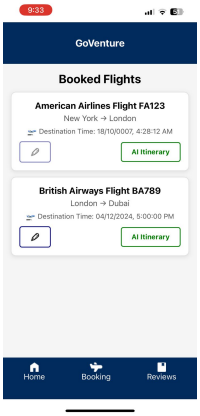
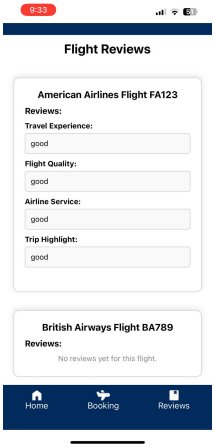
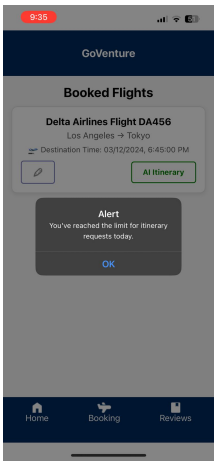


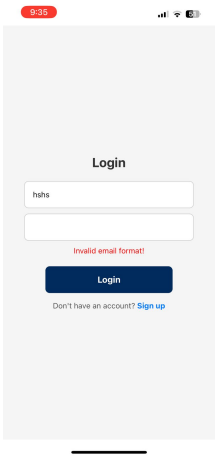
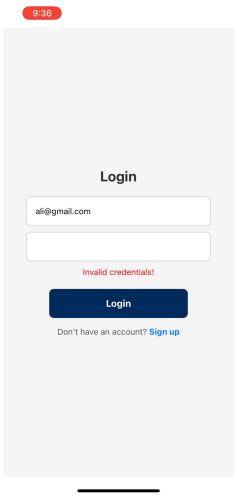


- Manual Testing

A number of manual tests were conducted, during the development phase and after it to allow the correct navigation of the users. This helped use the application from the perspective of the user, allowing the identification of other additional things in the app.

Test Case	Image

Ensure that the user is not allowed to book more that 5 tickets	
Ensure that the User is able to write a review	
Ensure that the review CTA is disabled when the user has already written a review	
Ensure that the user's limit for the AI itinerary is 2 per day	

Ensure that the login page prompts for an invalid email address	
Ensure that the login page prompts for invalid credentials	

6. Evaluation

The evaluation of the project consisted of both automated and manual components to access the reliability and usability of the application. The primary objective to judge how effectively the system met its purpose in terms of providing relevant recommendations as well as the user experience.

The manual usability evaluation was done through manual testing of the entire workflow, from registering into the application, all the way to writing a review, focussing on the was the application handled wrong data, mitigation for this involved error messages for input validation, CTA's and step by step modals so the user enters data and does not skip it. The goal of the testing was to determine whether the application provides clear guidance as well as maintains a consistent experience across different user actions.

For the AI itinerary section of the application, the evaluation was conducted using RAGAS to measure different metrics of the response and how well they aligned with the prompt provided and context given. The results were quantified to enable objective comparisons to highlight any weakness in the AI response quality that could end up affecting the user experience.

To support the performance, Postman was used to benchmark the response time and payload size when the CTA called the API. This helped assess how well the system performs in the real world scenario. To add to this, the API is hit very quickly but on the application the response generation takes up to 5 seconds, to avoid user aggression a loader has been placed on the screen till the response is shown.

Overall, the evaluation process was designed to assess the overall usability of the application. These softwares offer a comprehensive picture of how the system performs and areas of improvement.

7. Results

Manual testing confirmed the application delivers a smooth and intuitive user experience, The application guides users through key auctions, enforcing important restrictions, this contributes to both the functional integrity and usability.

The interactive elements respond appropriately to user behaviour thus enhancing the ease of use. Error handling is implemented to help users quickly identify their issues without any confusion. These findings suggest the application to be user friendly, responsive and well aligned with the expected requirements.

The results of the AI generated itineraries and the performance of the API integration were made using the RAGAS framework. Faithfulness scored 0.5556, this indicated that over half of the generated response was factually consistent with the provided content, however this is because the content was more descriptive and the goal of the response was supposed to be concise and in bullet points, thus this was expected. The contect precision score of 0.4405 shows that AI partially followed the expected output format but also contained imprecise information too. The answer relevancy was low at 0.0746 revealing that the response deviated from the users prompt, this was due to the buller point requirement.

On the performance side, the API testing conducted via Postman yielded an average response time of 1.43 seconds and a payload size of 2.15KB, this indicated an efficient request handling and lightweight data exchange.

8. Discussion

The evaluation of the application reveals several key insights about its performance, usability and limitations. The manual testing demonstrated areas of the workflow where incorrect data entry has been mitigated through the use of error messages and the disabled CTA's. This reflects thoughtful implementation of core usability principles however, the findings are constrained to a limited scope of testing and did not extend to automation. Additionally, the testing was conducted by myself through limited knowledge and thus it is possible that many edge cases remain uncertain.

From the testing and test result process, a number of constraints have been identified including the restriction on daily AI usage, which in this case limits the user to two itinerary generations per day. This can be perceived negatively by the users who wish to compare multiple destinations in one session however, the restriction has been placed to prevent API costs and overuse. Additionally, the lack of personalization in the AI output gives a very robotic effect to the application, neither does it make use of the exact destination dates to plan the itinerary based on the weather or local events. The current result is a one size fits all box.

Another limitation is the absence of offline functionality, the lack of this functionality makes users face difficulties if they want to regenerate the itinerary list on their vacation but due to this limitation will not be able to. Additional features include the absence of a retry button in case the API generates no response bringsing lack of trust in the application..

Other features of the application that could be enhanced is the use of real time flight APIs to give real information to the users, allowing them to be able to book flights and choose seats according to the current available options. Additionally, incorporating real time APIs like weather, transport or event listing ones could significantly increase the effectiveness of the application, thus providing a single application that books and provides additional features for users that want everything in one app.

In summary, the application provides a clean interface and stable core functionalities but for future developments, can be enhanced to focus on a scalable application that provides all the relevant details and features for users.

9. Conclusion

Throughout the course of the project, several core and advanced objectives were established with the goal of developing a user friendly application. Although many functionalities were met, some were left untouched.

One of the core objectives was to develop a user friendly application using React Native. The application does work across both Android and iOS platforms making sure that navigation is smooth. However, as mentioned above, some usability and personalized user experience aspects are missing from the application to further improve the experience.

Another significant objective was to implement a robust user authentication system. The features are functional and make use of the AsyncStorage to provide security but some features can also be added. For instance, a ‘Forgot Password’ feature could be added to streamline the process of the authentication as well as making use of Captcha or 2MFA to ensure that the user links their phone number for a secure application experience, enhancing the accessibility and reliability of the process.

The flight list shown on the HomeScreen is hardcoded data, although the integration of real time APIs was aimed for, the objective was not fully realized due to the timeframe and scale of the project. The application currently makes use of hard coded static data and limits the functionality and real world applicability. This gap ensures users are presented with accurate and timely flight information.

The booking functionality, although meets the objectives, could make use of additional features for the users to add other information such as baggage, seat class category and food option selections.

Advanced objectives such as the integration of real time flight updates and payment gateway usage were also part of the project scope but due to time constraints and the complexity of integrating third party services. Additionally, the application was designed with scalability in mind.

In summary, while significant progress was made to meet the core objectives of the application, some objectives were achieved at basic levels while others were difficult to realize into the application. Moving forward, the integration of advanced features and improving performance are to be focussed. The project highlights the importance of not only meeting the functional requirements but also ensuring the overall experience of a seamless user experience. The testing section helped to guide to future improvement and iterations of the application

10. Bibliography

Gündüz, F., 2020. Usability Improvements for Touch-Screen Mobile Flight Booking Application: A .

Ismail, A., 2016. iTourism Travel Buddy Mobile Application .

Ismail, M., 2022. Mobile Cloud Database Security: Problems and Solutions.

Kasinathan, V., 2017. Footprint: Tourism Information Search based on Mixed Reality .

Kuitunen, M., 2019. Cross-platform Mobile Application Development with React Native.

Londhe, K., 2023. Enhanced Travel Experience using Artificial Intelligence : A Data-driven Approach.

Martin, D., 2017. International Journal of Contemporary Hospitality Management.

Oleksandr, K., 2019. Modern State Web Application State Management and React Context Api Incorrect Usage for It.

Putrawan, Fahru Alfarizi Hananza , 2019. Design Thinking In UI/UX Lombok Halal Travel Application Project – Exploring The Expert And The Travel Industries User.

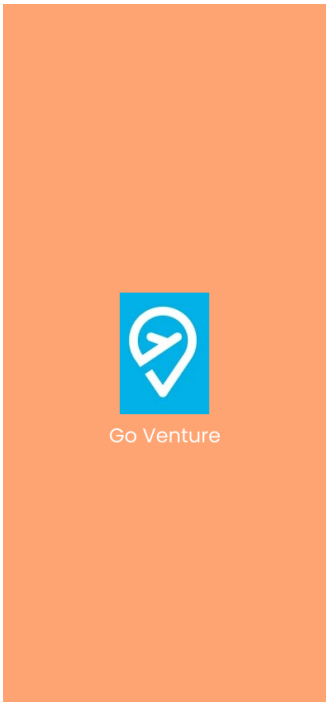
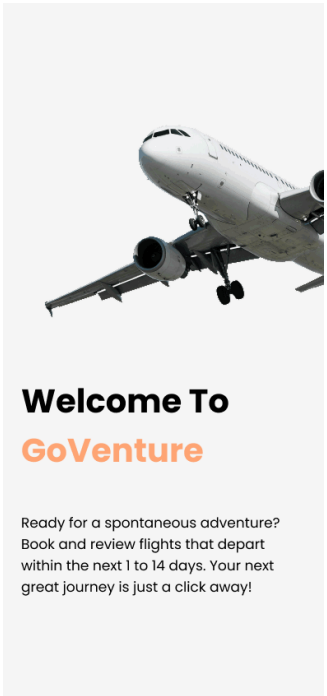
Ribeiro, F.R., 2018. Mobile applications for accessible tourism: overview, challenges and a proposed platform.

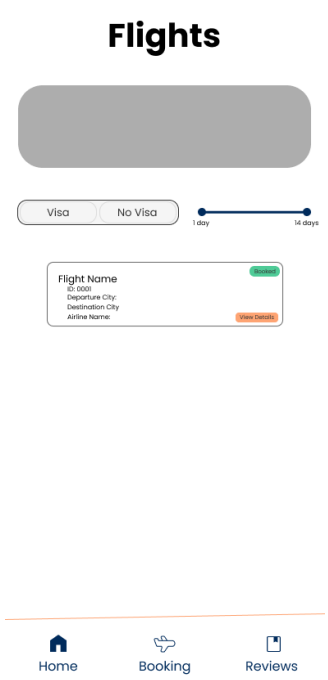
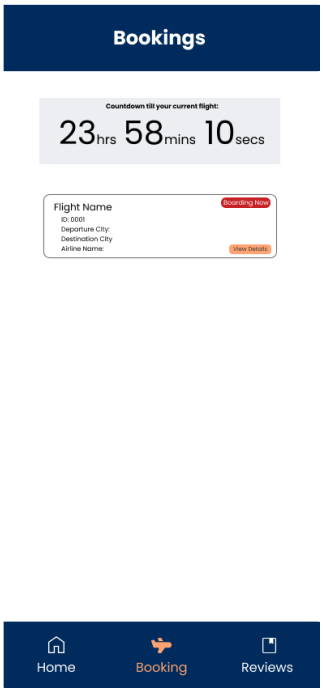
Tutunea, M.-F., 2019. The Generations` Preferences for Online Travel Booking Systems and Tools.

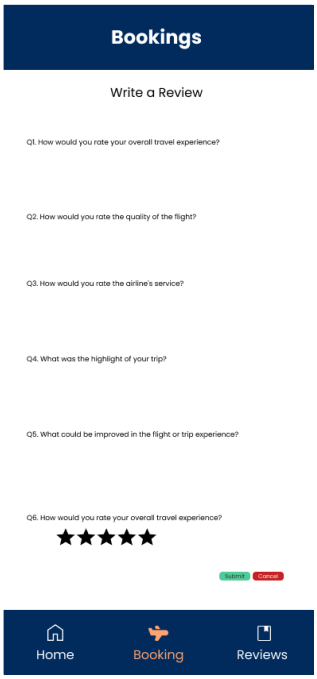
Yang, W., 2019. Security analysis of third-party in-app payment in mobile applications.

11. Appendices

11.1. Hi-fi designs

Splash Screen	
Welcome Screen	

Home Screen	
Booking Screen	

	
Review Screen	

11.2. RAGAS on the model

```
[2] !pip install -q ragas datasets openai tiktoken
    !pip install -q "ragas[metrics]"
```

```
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 187.2/187.2 kB 4.5 MB/s eta 0:00:00
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 491.2/491.2 kB 16.8 MB/s eta 0:00:00
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.2/1.2 MB 41.5 MB/s eta 0:00:00
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 116.3/116.3 kB 7.7 MB/s eta 0:00:00
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 45.5/45.5 kB 2.4 MB/s eta 0:00:00
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 183.9/183.9 kB 10.6 MB/s eta 0:00:00
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 143.5/143.5 kB 9.6 MB/s eta 0:00:00
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.5/2.5 MB 66.0 MB/s eta 0:00:00
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 61.3/61.3 kB 4.0 MB/s eta 0:00:00
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 194.8/194.8 kB 11.9 MB/s eta 0:00:00
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 50.9/50.9 kB 3.1 MB/s eta 0:00:00
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviour is the source of the following dependency conflicts.
torch 2.6.0+cu124 requires nvidia-cublas-cu12==12.4.5.8; platform_system == "Linux" and platform_machine == "x86_64", but you have nvidia-cublas-cu12 12.5.3.2 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cuda-cupti-cu12==12.4.127; platform_system == "Linux" and platform_machine == "x86_64", but you have nvidia-cuda-cupti-cu12 12.5.82 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cuda-nvrtc-cu12==12.4.127; platform_system == "Linux" and platform_machine == "x86_64", but you have nvidia-cuda-nvrtc-cu12 12.5.82 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cuda-runtime-cu12==12.4.127; platform_system == "Linux" and platform_machine == "x86_64", but you have nvidia-cuda-runtime-cu12 12.5.82 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cudnn-cu12==9.1.0.70; platform_system == "Linux" and platform_machine == "x86_64", but you have nvidia-cudnn-cu12 9.3.0.75 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cufft-cu12==11.2.1.3; platform_system == "Linux" and platform_machine == "x86_64", but you have nvidia-cufft-cu12 11.2.3.61 which is incompatible.
torch 2.6.0+cu124 requires nvidia-curand-cu12==10.3.5.147; platform_system == "Linux" and platform_machine == "x86_64", but you have nvidia-curand-cu12 10.3.6.82 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cusolver-cu12==11.6.1.9; platform_system == "Linux" and platform_machine == "x86_64", but you have nvidia-cusolver-cu12 11.6.3.83 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cusparse-cu12==12.3.1.170; platform_system == "Linux" and platform_machine == "x86_64", but you have nvidia-cusparse-cu12 12.5.1.3 which is incompatible.
torch 2.6.0+cu124 requires nvidia-nvjitlink-cu12==12.4.127; platform_system == "Linux" and platform_machine == "x86_64", but you have nvidia-nvjitlink-cu12 12.5.82 which is incompatible.
gcsfs 2025.3.2 requires fsspec==2025.3.2, but you have fsspec 2024.12.0 which is incompatible.
WARNING: ragas 0.2.14 does not provide the extra 'metrics'
```

```
[3] import openai
    import os
    from ragas import evaluate
    from ragas.metrics import answer_relevancy, faithfulness, context_precision
    from datasets import Dataset
```

```
[8] from openai import OpenAI

    # ✅ Create a client instance
    client = OpenAI(api_key="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")

    def generate_itinerary(destination):
        response = client.chat.completions.create(
            model="gpt-4o-mini",
            messages=[
                {
                    "role": "system",
                    "content": "You are a travel assistant that generates short, concise lists of places to visit with no additional descriptions or details. Only provide the names of the places i
                },
                {
                    "role": "user",
                    "content": f"I am going to {destination}. Please give me a list of places to visit, no descriptions, in bullet points. Only the names of the places, 5-10 locations max."
                }
            ]
        )

        return response.choices[0].message.content.strip()
```

```
[9] destination = "London"
    generated_answer = generate_itinerary(destination)
    print("✅ Generated Answer:\n", generated_answer)
```

```
✅ Generated Answer:
 - The British Museum
- Tower of London
- Buckingham Palace
- The Houses of Parliament
- The Tate Modern
- Covent Garden
- The London Eye
- Westminster Abbey
- Camden Market
```

```
[13] import os
     os.environ["OPENAI_API_KEY"] = "sk-proj-PDSPnULdc9RhUIfF1cP_kVjI-Je1UH8Ca3NbHTW6CmYI7MASxWesclwzn054uth9z1MnBUcJrqT3BlbkFJmop3tA3TucPkQw8CaRuqJRU0dQZLcGqOzWoQRvHufBFD72kkOV6uam4mWl9-lVmUrXJ3kH
```

```
[10] reference_answer = """
     - Buckingham Palace
     - Tower of London
     - British Museum
     - London Eye
     - Hyde Park
     - Big Ben
     - St Paul's Cathedral
     - Camden Market
     """


     retrieved_context = [
         "Buckingham Palace is the official residence of the British monarch.",
         "The London Eye is a large observation wheel offering panoramic views of London.",
         "The British Museum holds a vast collection of world art and artifacts.",
         "The Tower of London is a historic fortress and former royal residence.",
         "Big Ben is a famous London clock tower.",
         "Hyde Park is a major park in Central London.",
         "Camden Market offers crafts, clothing, and street food.",
         "St Paul's Cathedral is a historic Anglican cathedral in London."
     ]
```

```
[11] data = {
         "question": [f"I am going to {destination}. Please give me a list of places to visit."],
         "contexts": [retrieved_context],
         "answer": [generated_answer],
         "reference": [reference_answer],
     }

     dataset = Dataset.from_dict(data)
```

```
[14] result = evaluate(dataset, metrics=[faithfulness, answer_relevancy, context_precision])
```

```
Evaluating: 100% ━━━━━━━━━━━━━━━━━━━━ 3/3 [00:14<00:00, 4.99s/it]
```

```
print("Evaluation Metrics:")
print(result)
```

```
Evaluation Metrics:
{'faithfulness': 0.5556, 'answer_relevancy': 0.8746, 'context_precision': 0.4405}
```