Report Draft

# Introduction

As the tourism industry grows rapidly, online tour booking has become essential for tourists and operators. To address this, we are developing a Tour Booking Management System using Agile methodology. This system will help users quickly search and book tours while providing operators with tools to manage their offerings. The frontend will use HTML, CSS, and JavaScript, while Python will handle backend logic. Two-week sprints allow flexibility in meeting user requirements.

The goal of this project is to create a user-friendly platform for tourists to find tour information, book, pay, cancel reservations, and leave reviews. Operators and administrators will have tools to manage tours, user accounts, generate reports, and monitor system performance to ensure efficiency.

The project focuses on New Zealand tours, including browsing information, online booking, reviews, and monitoring. The development will be divided into three versions, gradually adding features at each stage.

**Key Features:**

* **Tour Browsing:** View detailed information (description, price, schedule, ratings).
* **Manage Online Booking:** Secure booking, email confirmations, real-time status updates.
* **Tour Review:** Tourists can rate and review tours.
* **Admin Interface:** Manage user accounts, generate reports, and monitor system performance.
* **System Monitoring & Reporting:** Centralized performance and error log tracking.
* **Customer Support:** Email support for inquiries.
* **System Integration:** API for third-party integrations.

The main goal is to deliver a comprehensive and user-friendly online booking platform to meet both tourists' needs and enhance operators' efficiency.

# Team Formation

# Project Kickoff Meeting

# Requirement Gather Meeting

# Requirement Analysis and Prioritisation

# Agile Development Sprints

# Cost Estimation

# Acceptance Criteria and Testing

# Reflection Report

The Tour Booking System is a project that has its origin from the increasingly felt need in the tourism market for an effective, user-friendly online booking system targeting tours around New Zealand specially focus on Auckland.In this group project, we have worked toward developing a system that would give tourists an enhanced feeling during booking on one hand and operators a set of tools badly needed to operate tours with maximum efficiency on the other hand. Agile methodology allowed us to adapt flexibly in the face of evolving requirements and user feedback during all stages of the project's lifetime. This report reflects on the stages of demand gathering, requirement analysis, code implementation, testing, and release, focusing on the lessons learned at each stage. Here are the following details of the reflection for each stages.

**Demand Gathering**

This is the demand-gathering phase, during which we have had limited stakeholder interactive engagement in the form of surveys and interviews with potential users: tourists, tour operators, and administrators. Of course these roles are acted by each of our team member respectively and virtually. Role of tourists had a number of suggestions on how to make their booking simpler, having full information about the tour, and reviewing options. One important challenge was to strike a balance between the seemingly conflicting requirements presented by different needs, and these needs were carefully documented and categorized and had to be balanced.

**Requirements Analysis**

We prioritized requirements into features as "Must-have," "Should-have," "Could-have," and "Won't-have." The reason for this methodology is that it will enable us to stick to core functionalities while still accommodating some functionalities that can be part of future enhancement plans. Hence, at this stage, the user demands were mapped into functional and non-functional requirements. This helped us outline the project scope for each version and helped plan our Agile sprints in that respect.

**Coding**

The coding was divided into two-week sprints in which we would develop specific features based on the prioritized requirements. We went with Agile methodologies which allowed us to fix our development process as we went and to perform feedback intake or any other changes on the go right away. After each sprint, we had a working prototype, reflecting the incremental approach towards our product in its entirety.

**Testing**

Testing forms one of the very key features related to the project, being done through various stages of the project in order to ensure the quality of the system. We started unit testing, verifying that all the components were behaving as anticipated. Then we moved on to integration testing, and following UAT. But due to the time limitation we skipped performance test in the first release and planned to implement in the second stage for the reason that at the earlier period of online services, the number of registered costumers will be less.

**Challenges and Lessons Learnt**

We faced various types of challenges while working on this project; each taught us something important. An important challenge was coordination and communication within the team during different phases of the project. The overall cooperation among each one of the team members taught us the importance of clear communication protocols and collaboration tools, which maintained the alignments and reduced misunderstandings.

Other challenges were technical integrations because rigorous testing and fixing needed to be done to ensure a smooth flow of data across these components, especially when real-time updates occurred in the booking status. This only reinforced the idea of modular and robust documentation since these allowed us to fix problems much faster.

Finally, the time management was an issue, too, especially in the closing stages after the coding phases. In balance with the thoroughness of testing, time-wise for the project forced us to prioritize which features were most vital and which we could get away with not focusing on for each sprint. This taught us the importance of setting realistic goals and timelines for each cycle of development, since project timelines have to be respected-because one wants to deliver a quality product within expected times.

The challenges we have gone through really imparted lessons on communication, technical integration, time management, and user-centered development. Such insight from our group project will surely help us to develop anything even more effectively and efficiently in our future projects.

1. Scope of Work

The system focuses on providing tour booking services exclusively for tours within New Zealand. It will cover the following key features:

1. **Tour Browsing**: Displays detailed information such as tour descriptions, prices, itineraries, and ratings.
2. **Online Booking**: Supports email confirmation, payment processing, and tracking of booking statuses.
3. **Tour Review**: Allows users to provide feedback and ratings after the tour.
4. **User Management**: Enables administrators to manage user accounts and permissions.
5. **System Integration**: Offers APIs for integrating with third-party payment services and tour information providers.

**Out of Scope**

The following items are beyond the scope of this project:

1. Offline travel arrangements or services outside of New Zealand.
2. **Travel insurance management**: The system will not handle or process insurance requests or claims.
3. **Multi-currency support**: The platform will only process payments in New Zealand dollars (NZD) without exchange rate calculations.
4. System Design

**System Architecture**

1. **Frontend Technologies:** The frontend will be developed using HTML, CSS, and JavaScript, ensuring a responsive and user-friendly interface for tourists and administrators.
2. **Backend Technologies:** The backend logic will be handled using a combination of Python and Node.js to efficiently manage business logic, API calls, and data processing.
3. **Database:** The system will use SQLite as the primary database to store data such as tour details, user information, bookings, and feedback.

**Module Breakdown**

1. **Tour Browsing Module:** Loads available tours from the database and displays relevant information to the users based on their preferences.
2. **Online Booking Module:** Processes bookings and payments, updates booking status, and handles email confirmations.
3. **Admin Interface:** Provides system administrators with the ability to manage users, process booking requests, and generate reports.
4. **Tour Review Module:** Collects and displays feedback from users, including ratings and comments.
5. **System Integration Module:** Integrates with third-party services through APIs, such as payment gateways.

**System Interfaces**

1. **Tour Listing Page:** Displays all available tours and allows users to filter based on preferences like destination, date, and tour type.
2. **User/Admin Login Page:** Provides separate login interfaces for tourists and administrators.
3. **Booking Page:** Facilitates the booking process by collecting traveler details, confirming availability, and processing payments.

**Data Flow and Communication**

1. **Frontend and Backend Communication:** The frontend will communicate with the backend using REST APIs to ensure smooth data exchange. Node.js will handle API requests, while Python will manage the business logic and data processing.
2. **Database Operations:** SQLite will store all necessary data, including user accounts, tours, bookings, and reviews. The backend will manage data queries and ensure synchronization between modules.
3. **Scalability Considerations:** Although SQLite is a lightweight database suitable for the initial development phase, the system architecture allows for future migration to more robust databases (e.g., MySQL or PostgreSQL) if needed.