Project Proposal: Travel Agency Management System

Peter Jiang

April 29, 2025

1 Introduction

This proposal outlines the development of a Travel Agency Management System. The system aims to streamline operations for travel agencies by providing a platform for managing travel bookings, itineraries, and customer information. The system will employ Object-Oriented Programming (OOP) principles, utilize file I/O for data persistence, and feature a user-friendly graphical interface built with Java Swing.

2 Real-World Scenario and Business Logic

2.1 Scenario Overview

The Travel Agency Management System serves the needs of a modern travel agency that offers various travel packages to different destinations. The agency handles customer bookings, maintains trip itineraries, manages payments, and collects customer reviews. This system will digitize these processes, enhancing efficiency and customer satisfaction.

2.2 Business Logic

The core business logic of the Travel Agency Management System includes:

- **Customer Management**: Register new customers, update customer information, and track customer preferences and booking history.
- **Package Management**: Create, update, and delete travel packages with details such as destination, duration, accommodations, activities, and pricing.
- **Booking Process**: Allow customers to browse packages, make reservations, and process payments.
- **Itinerary Management**: Create and modify detailed day-by-day itineraries for each package or custom trip.
- **Review System**: Collect and display customer reviews and ratings for travel packages.
- **Reporting**: Generate reports on sales, popular destinations, and customer satisfaction metrics.

3 Object-Oriented Design

3.1 Class Hierarchy

The system will implement a robust class structure that leverages key OOP concepts including inheritance, polymorphism, interfaces, and abstract classes.

3.2 UML Class Diagram

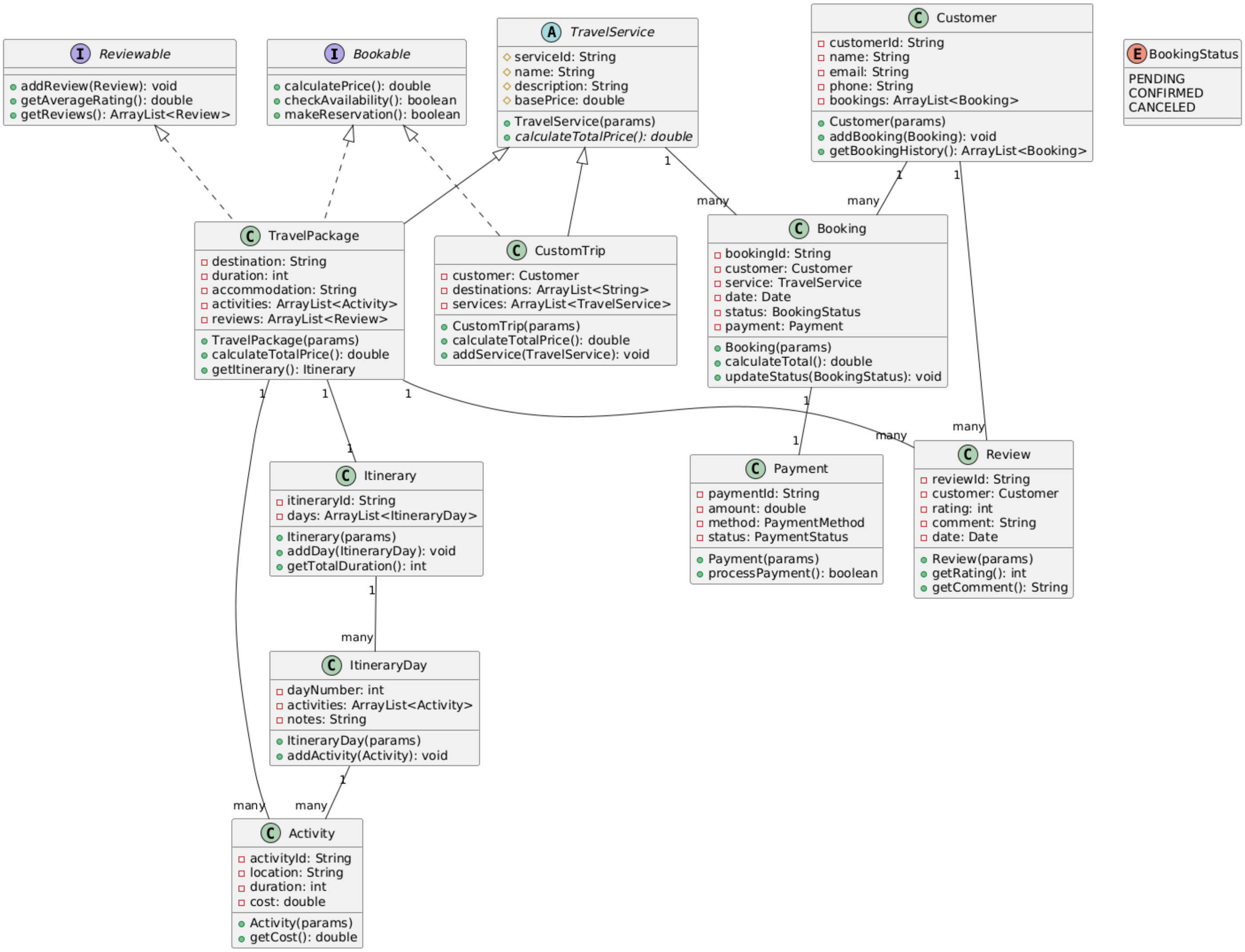


Figure 1: UML Class Diagram for Travel Agency Management System

3.3 OOP Concepts Implementation

3.3.1 Interfaces

The system implements two main interfaces:

- **Bookable**: Defines methods that any bookable service must implement, including `calculatePrice()`, `checkAvailability()`, and `makeReservation()`.
- **Reviewable**: Defines methods for entities that can receive customer reviews, including `addReview()`, `getAverageRating()`, and `getReviews()`.

3.3.2 Abstract Classes

- **TravelService**: An abstract class that provides common attributes and methods for all travel services. It includes abstract methods like `calculateTotalPrice()` that must be implemented by subclasses.

3.3.3 Inheritance

The inheritance hierarchy includes:

- `TravelService` serves as the parent class for `TravelPackage` and `CustomTrip`.
- Each subclass inherits common attributes and methods while implementing specific behaviors.

3.3.4 Polymorphism

Polymorphism is demonstrated through:

- Different implementations of `calculateTotalPrice()` in `TravelPackage` and `CustomTrip`.
- Use of the `Bookable` interface allowing different bookable services to be treated uniformly.

3.3.5 Method Overloading

- Multiple constructors in classes like `Booking` and `Customer` with different parameter sets.
- `addService()` method in `CustomTrip` with versions for adding single services or collections.

3.3.6 Method Overriding

- Override of `calculateTotalPrice()` in subclasses of `TravelService`.
- Override of `toString()` in various classes for customized string representation.

3.3.7 Exception Handling

Custom exceptions will include:

- `BookingException`: For handling errors in the booking process, such as unavailable packages.
- `PaymentProcessException`: For managing payment processing errors.

4 Graphical User Interface Design

The GUI will be implemented using Java Swing, providing an intuitive interface for users to interact with the system.

4.1 GUI Structure (Booking Panel)

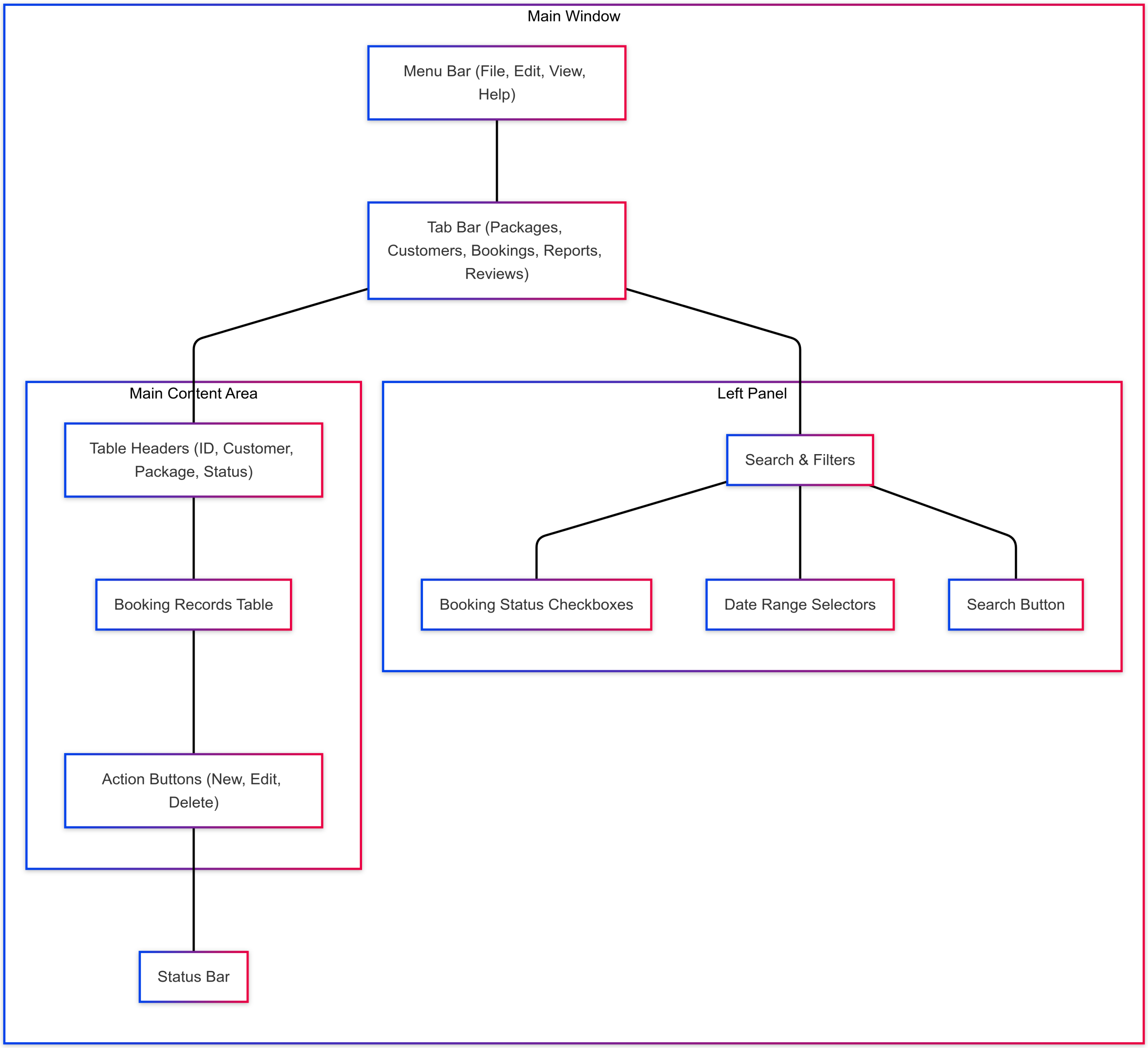
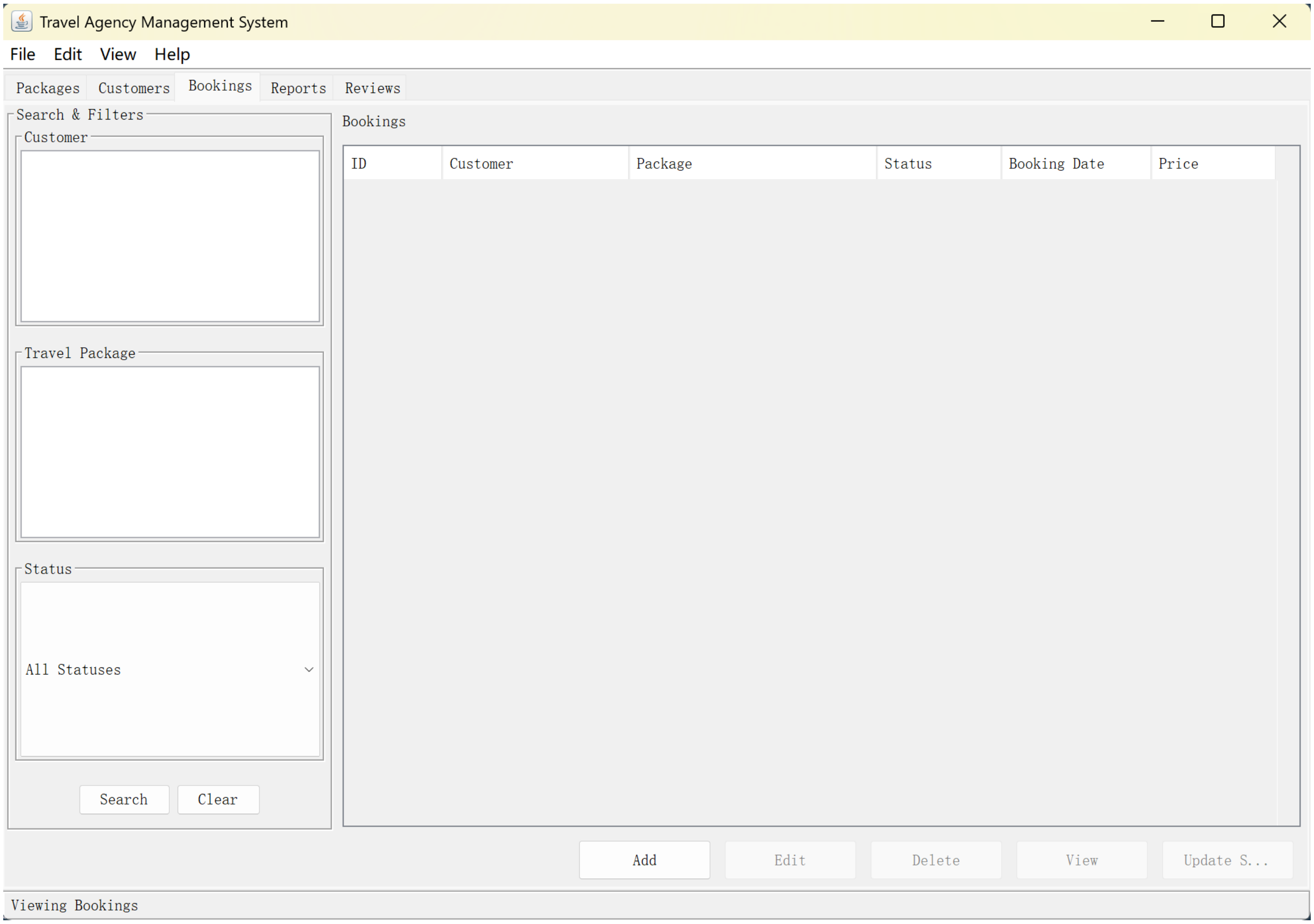

Main Window
Menu Bar (File, Edit, View, Help)
Tab Bar (Packages, Customers, Bookings, Reports, Reviews)
Main Content Area
Table Headers (ID, Customer, Package, Status)
Booking Records Table
Action Buttons (New, Edit, Delete)
Left Panel
Search & Filters
Booking Status Checkboxes
Date Range Selectors
Search Button
Status Bar

Figure 2: GUI Structure (Booking Panel)



ID	Customer	Package	Status	Booking Date	Price

Status: All Statuses

Viewing Bookings

Figure 3: Booking Panel

4.2 GUI Features

The GUI will include the following features:

- **Tab-based Navigation**: Easy navigation between different system functions (Packages, Customers, Bookings, Reports, Reviews).
- **Search and Filtering**: Ability to search and filter data across all modules.
- **CRUD Operations**: Interface elements for creating, reading, updating, and deleting records:
  - **Create**: Forms to add new packages, customers, bookings, etc.
  - **Read**: Tables and detailed views to display information.
  - **Update**: Forms to modify existing records.
  - **Delete**: Options to remove records with confirmation.
- **Reports Section**: Visual representation of sales data, popular packages, and customer reviews.
- **Notification System**: Alerts for important events like booking confirmations or payment processing.

5 Data Storage Design

5.1 File I/O Architecture

The system will use file I/O to store and retrieve data persistently. The data storage will be organized as follows:

- **JSON File Format**: Data will be stored in JSON format for readability and ease of handling.
- **File Structure**:
  - `customers.json`: Customer information and preferences.
  - `packages.json`: Travel package details and availability.
  - `bookings.json`: Booking records and status.
  - `reviews.json`: Customer reviews and ratings.
  - `activities.json`: Available activities for packages.
- **Data Loading**: When the application starts, it will load data from these files into memory.
- **Data Saving**: Changes will be saved back to the files when data is modified or when the application closes.
- **Backup Mechanism**: The system will create backups of data files periodically to prevent data loss.

5.2 Data Processing

The system will include:

- **Data Validation**: Validating user inputs before saving to ensure data integrity.
- **Sorting and Searching**: Implementation of algorithms to efficiently sort and search through data collections.
- **Data Analysis**: Functions to analyze booking trends, popular destinations, and revenue statistics.

6 Additional Features

6.1 Advanced Search Functionality

The system will include advanced search options to:

- Search packages by destination, date range, price range, or activities.
- Filter customer records by booking history or preferences.
- Search bookings by status, date, or package type.

6.2 Data Analysis

The system will offer data analysis features:

- Monthly and yearly booking trends.
- Most popular destinations and packages.
- Customer spending patterns.
- Revenue forecasting based on booking data.

7 Conclusion

The proposed Travel Agency Management System will provide a comprehensive solution for travel agencies to manage their operations efficiently. By implementing the outlined features and following object-oriented design principles, the system will offer a robust, user-friendly platform that enhances the agency's ability to serve customers and track business performance.

The development will focus on creating a modular, maintainable system that fulfills all the requirements specified in the project guidelines, including OOP concepts, data storage, GUI implementation, and additional features for data analysis and advanced search functionality.