**Airline Ticket Booking Requirements Specification**

**Version 1.0**

**May, 2025**

**Group Members:**

Anxhela Kullafi

Frenki Zelo

Kledisa Collaku

Martin Shameti

Megi Selimi

**Table of Contents**

1. EXECUTIVE SUMMARY1

1.1 Project Overview2

1.2 Purpose and Scope of this Specification2

2. PRODUCT – SERVICE DESCRIPTION4

2.1 Product Context2

2.2 User Characteristics2

2.3 Assumptions2

2.4 Constraints2

2.5 Dependencies2

3. REQUIREMENTS4

3.1 Functional Requirements5

3.2 Non - Functional Requirements5

3.2.1 Product Requirements6

3.2.1.1 User Interface Requirements6

3.2.1.2 Usability6

3.2.1.3 Efficiency6

3.2.1.3.1 User Interface Requirements6

3.2.1.3.1 Usability6

3.2.1.4 Dependability6

3.2.1.5 Security6

3.2.2 Organizational Requirements6

3.2.2.1 Environmental Requirements 6

3.2.2.2 Operational Requirements 6

3.2.2.3 Development Requirements 6

3.2.3 External Requirements6

3.2.3.1 Regulatory Requirements 6

3.2.3.2 Ethical Requirements 6

3.2.3.3 Legislative Requirements 6

3.2.3.3.1 Accounting Requirements6

3.2.3.3.1 Security Requirements6

3.3 Domain Requirements5

4. USER SCENARIOS – USE CASES4

4.1 Major Functions2

4.2 Use Cases2

5. DIAGRAMS4

5.1 Use Case Diagram2

5.2 Activity Diagrams2

5.3 ERD Diagram2

5.4 Class Diagram2

5.5 Component Diagram2

5.5 Sequential Diagram2

1. **EXECUTIVE SUMMARY**

The project involves the development of a dedicated airline ticket booking software designed specifically for a single airline. The system will allow passengers to search, filter, book flights, select seats, and make payments securely. It also provides administrative tools for managing flights, monitoring bookings, and a dedicated tech support user for system maintenance.  
The software will focus on a user-friendly, secure, and efficient experience for both passengers and administrators, resembling modern airline booking systems.

* 1. **Project Overview**

This project focuses on creating an all-in-one airline ticket booking software aimed at simplifying the flight reservation process for passengers while offering powerful management tools for airline staff. The software will feature a range of functionalities such as ensuring seamless booking, secure payment handling etc.

**Intended Audience:**

* **Passengers:** Regular users booking flights.
* **Administrators:** Airline staff managing the flight and booking data.
* **Tech Support:** Personnel responsible for the software's technical health.

It will serve the following main purposes:

**Passengers** can:

* Register an account with their email.
* Log in and manage their account.
* Search and filter flights (by destination, date, number of persons, class).
* Select flights, choose seats, and complete payments securely.
* Receive booking confirmations via email.

**Administrators (Managers)** can:

* Add, modify, and delete flight details.
* Manage and oversee passenger bookings.
* Adjust ticket prices, flight schedules, and seat availability.
* View reports on flight occupancy, revenue, and performance.

**Tech Support / Maintenance**:

* A single responsible user tasked with ensuring the system remains operational, updating the software, and fixing any detected bugs.

The system will **not cover** additional airport services such as baggage handling, cargo logistics, or multi-airline management — focusing solely on **one airline**'s booking and internal management processes.

* 1. **Purpose and Scope of this Specification**

The purpose of this document is to outline the requirements for an airline ticketing system. This system aims to provide seamless booking, payment, and management of airline tickets for customers while allowing airline administrators to oversee operations efficiently. Additionally, software maintenance technicians will be able to ensure the system runs smoothly by addressing any technical issues.

This specification is intended to:

* Serve as a guideline for the **development team**.
* Set **expectations** for the **end users** (passengers and admins).
* Act as a **baseline** for quality assurance and project validation.

The scope of this specification, the airline ticketing system will enable users to search for flights, book tickets, and manage their travel plans. Airline administrators will be responsible for managing flight schedules, ticket availability, and customer bookings. The software maintenance technician will handle system maintenance, updates, and troubleshooting.

* In Scope

This specification covers functionalities related to:

**Passenger Services:**

* Account registration, login, and profile management.
* Flight search with filters (destination, date, number of persons, seat class).
* Flight selection and seat reservation.
* Payment processing (via credit/debit card).
* Booking confirmation emails.

**Administrator Services:**

* Flight creation, modification, and deletion.
* Price and schedule management.
* User booking management and reporting tools.

**Tech Support Services:**

* System maintenance and updates.
* Bug tracking and resolution.
* Out of Scope

Management of **airport operations** (baggage, gates, lounges).

**Multi-airline integration** (system is for **one airline only**).

**Third-party** system integrations not explicitly described (e.g., external payment providers unless specified).

**Future system expansions** (such as hotel booking, rental car services) are **excluded** from the current scope and may be considered for future phases.

1. **PRODUCT – SERVICE DESCRIPTION**

## **Background Information**

This airline ticket booking software is tailored to meet the dynamic demands of the airline industry and its passengers. With the travel sector rapidly embracing digital transformation, there is an increasing need for streamlined, intuitive, and secure booking systems. The software seeks to improve passengers' travel experiences while equipping airline teams and departments with powerful management tools.

Several key factors have shaped the development and requirements of this software:

* **User Experience:**
  + Passengers expect a simple and intuitive booking experience.
  + A clear interface for searching flights, selecting seats, and processing payments is critical.
  + Accessibility across different devices (PCs, tablets, smartphones) is important.
* **Security:**
  + Protecting user data, especially payment information, is a top priority.
  + Secure authentication and encryption mechanisms are required.
* **Real-Time Information:**
  + Passengers should see live updates about available flights, prices, and seating.
  + Flight changes or cancellations must be reflected instantly.
* **Operational Efficiency:**
  + The administrator/business manager needs efficient tools to manage flights, prices, and monitor booking trends.
  + Financial summaries and reporting features help with airline management decisions.
* **System Reliability and Maintenance:**
  + The tech support/maintenance team requires robust tools for system monitoring, updates, and troubleshooting.
  + Fast recovery in case of system issues ensures high service availability.
* **Customer Support and Notifications:**
  + Automated emails should notify passengers about booking confirmations, schedule changes, and other important updates.
  + Ensuring passengers stay informed reduces miscommunication and increases trust.

By addressing these factors, the platform ensures a **high-quality experience** for all users while maintaining strong security and operational efficiency.

* 1. **Product Context**

The airline ticket booking software can function as an independent, self-contained system with its primary purpose being to facilitate the booking of airline tickets for passengers.

The airline ticket booking system is an **independent**, **self-contained** platform focused solely on the **flight booking activities** for a **single airline**.  
The core operations — including flight browsing, booking, payment, and notification — are managed entirely within the system.

However, to achieve a complete service offering, the system interfaces with the following external components:

* **Payment Gateway:**
  + Integration with a secure online payment processor for handling passenger transactions.
  + Supports debit cards, credit cards, and other secure digital payment options.
* **Email Notification System:**
  + Automated emailing system to send booking confirmations, payment receipts, and flight updates to passengers.
* **Internal Database:**
  + Secure storage of flight schedules, seat availability, passenger accounts, booking records, and payment history.

### Summary of System Relationships:

* **Independent Capabilities:**
  + Complete booking cycle (flight selection, seat selection, payment, confirmation) managed internally.
* **Integrated Components:**
  + Secure Payment Processor (for financial transactions).
  + Email Notification System (for customer communications).

The system does **not** integrate with Global Distribution Systems (GDS) or third-party travel agencies, as it is intended to serve only **one airline’s direct customers**.

* 1. **User Characteristics**

The system serves **three main types of users**, each with different expectations and technical abilities:

### **1. Passengers**

* **Experience:**
  + Range from occasional travelers to frequent flyers.
  + Basic to moderate experience using digital booking systems.
* **Technical Expertise:**
  + Familiar with web browsing and mobile applications.
  + Expect clear navigation and fast performance.
* **General Characteristics:**
  + Prefer quick search, easy booking steps, and immediate confirmation.
  + Appreciate clear display of prices, baggage rules, and seat options.
  + Rely on timely emails for booking and schedule updates.

### **2. Administrator / Business Manager**

* **Experience:**
  + Skilled in airline operations, booking management, and financial oversight.
* **Technical Expertise:**
  + Comfortable working with management dashboards and reporting tools.
  + Ability to create and modify flight schedules, manage pricing, view booking statistics.
* **General Characteristics:**
  + Require access to real-time booking data and financial summaries.
  + Need tools for adjusting flight schedules and prices quickly and securely.
  + Prioritize system reliability and operational transparency.

### **3. Tech Support / Maintenance**

* **Experience:**
  + Specialized in IT systems, web server maintenance, and technical troubleshooting.
* **Technical Expertise:**
  + Advanced knowledge of backend systems, databases, software security, and server monitoring.
* **General Characteristics:**
  + Responsible for ensuring system uptime, applying updates, and fixing bugs.
  + Expected to monitor the system continuously, handle backups, and perform emergency maintenance.
  + Must ensure that data security and regulatory compliance are maintained.
  1. **Assumptions**

**Assumptions Affecting Requirements**

**1. Equipment Availability**

* Assumption:  
  Users have access to modern computers, tablets, or smartphones.
* Impact:  
  The system must be responsive and compatible with various device types and screen sizes.
* Change Needed if Unavailable:  
  A simplified, lightweight version of the system would need to be developed for older devices.

**2. Operating System**

* Assumption:  
  Users are operating up-to-date versions of Windows, macOS, iOS, and Android.
* Impact:  
  The system should be tested against recent and several previous OS versions.
* Change Needed if Unavailable:  
  Expanded compatibility testing for older operating systems would be necessary.

**3. Internet Access**

* Assumption:  
  Users have stable and reliable internet access.
* Impact:  
  The system will be designed as a web-based platform requiring continuous internet connectivity.
* Change Needed if Unavailable:  
  Offline features or a downloadable mobile application might need to be considered.

**4. Browser Compatibility**

* Assumption:  
  Users utilize modern web browsers (e.g., Chrome, Firefox, Safari, Edge).
* Impact:  
  The platform will be optimized for these browsers.
* Change Needed if Unavailable:  
  Additional browser testing and fallback support for older browsers would be required.

**5. Technical Expertise**

* Assumption:  
  Users have a basic familiarity with online booking systems.
* Impact:  
  The user interface will focus on ease-of-use but assumes basic web navigation knowledge.
* Change Needed if Unavailable:  
  A more guided user experience (e.g., tutorials, onboarding help) would need to be integrated.

**6. Payment Methods**

* Assumption:  
  Users have access to digital payment methods such as debit/credit cards and digital wallets.
* Impact:  
  Integration with standard payment gateways is sufficient.
* Change Needed if Unavailable:  
  Manual payment options or support for alternative payment methods would be required.

**7. Airline Reservation System Integration**

* Assumption:  
  The system will primarily manage its own flight inventory without needing external GDS (Global Distribution Systems) integration.
* Impact:  
  Simplified internal flight management and booking processes.
* Change Needed if Unavailable:  
  External data source integration would need to be developed if broader flight options become necessary.

**8. Security Standards**

* Assumption:  
  Industry-standard security protocols (SSL/TLS encryption, PCI-DSS for payments) will be implemented.
* Impact:  
  Ensures user data protection and secure transactions.
* Change Needed if Unavailable:  
  Custom security protocols or enhanced compliance checks would be needed.

**9. Regulatory Compliance**

* Assumption:  
  The system complies with GDPR and local aviation industry regulations.
* Impact:  
  Maintains user trust and legal compliance.
* Change Needed if Unavailable:  
  Significant modifications to data handling and privacy mechanisms would be required.

**Summary**

These assumptions directly influence the software's design, features, and security. If any assumption proves invalid, the system requirements will need to be revised to maintain functionality, usability, and compliance.

* 1. **Constraints**

**Design Constraints**

**1. Parallel Operation with Existing Systems**

* Description:  
  The booking platform may need to operate alongside limited legacy systems during initial rollout.
* Impact:
* Data Synchronization:  
  Accurate syncing between old and new booking records.
* Compatibility:  
  New system must accept data formats used by legacy systems.
* User Training:  
  Clear training must be provided for users interacting with both systems temporarily.

**2. Audit Functions (Audit Trail, Log Files, etc.)**

* Description:  
  Comprehensive tracking of all significant system events and user actions.
* Impact:
* Data Logging:  
  Track creation, modification, cancellation of bookings with timestamps and user IDs.
* Security:  
  Logs must be tamper-proof and accessible only to administrators.
* Performance:  
  Logging should not interfere with booking speeds or overall system performance.
* Compliance:  
  Meet regulations on auditability and data retention.

**3. Access Management and Security**

* Description:  
  Robust security controls must be implemented.
* Impact:
* Authentication:  
  Multi-factor authentication for admin and critical functions.
* Role-Based Access Control:  
  Different permissions for passengers, administrators, and support personnel.
* Encryption:  
  Protect sensitive data both at rest and in transit.
* Monitoring:  
  Detect unauthorized access attempts and log security events.
* Compliance:  
  Meet GDPR and other relevant data protection laws.

**4. Criticality of the Application**

* Description:  
  The application is mission-critical for airline operations and customer trust.
* Impact:
* High Availability:  
  Redundant servers and failover plans to minimize downtime.
* Performance:  
  Ability to handle peak loads during promotions or peak seasons.
* Backup and Recovery:  
  Regular automated backups and fast disaster recovery mechanisms.

**5. System Resource Constraints**

* Description:  
  Limited hardware or cloud infrastructure budget initially.
* Impact:
* Optimization:  
  Efficient use of server resources, memory, and storage.
* Scalability:  
  Easy scalability as passenger volumes increase.
* Hardware Compatibility:  
  Must work with existing hosting or cloud platforms initially.

**6. Other Design Constraints**

* Programming Language / Framework:

System will be developed using modern frameworks like ASP.NET Core or Django.

* Standards and Guidelines:
  + - Follow web accessibility (WCAG) and usability best practices.
    - Comply with secure coding guidelines and API standards.
* Integration Standards:

APIs and external systems must meet secure integration requirements.

**Summary**

These constraints influence technical design, security practices, and deployment strategies. Managing these constraints carefully is critical for delivering a secure, user-friendly, and high-performance ticket booking system.

* 1. **Dependencies**

**Key Dependencies Affecting Requirements**

**1. Payment Gateway Integration**

* Dependency:  
  Successful integration with secure online payment gateways (e.g., Stripe, PayPal).
* Impact:
* Security and Compliance:  
  Must ensure PCI-DSS compliance.
* Testing:  
  Thorough end-to-end testing of all payment flows is mandatory.
* User Experience:  
  Seamless and trustworthy checkout process for users.

**2. Email Notification System**

* Dependency:  
  Integration with a reliable email service provider (e.g., SendGrid, AWS SES).
* Impact:
* Notification Delivery:  
  Confirmations, flight changes, and other critical updates must be sent instantly.
* Reliability:  
  High deliverability rates and email tracking are important for customer satisfaction.

**3. Hosting and Infrastructure**

* Dependency:  
  Availability of reliable web hosting or cloud services (e.g., AWS, Azure).
* Impact:
* Scalability and Reliability:  
  Infrastructure must support high traffic without failures.
* Monitoring:  
  Proactive server monitoring to detect and address issues quickly.

**4. Regulatory Compliance**

* Dependency:  
  Adherence to GDPR, local aviation regulations, and cybersecurity standards.
* Impact:
* Data Handling:  
  Secure, compliant storage and processing of user data.
* Consent Management:  
  Proper mechanisms for obtaining user consent for data usage.
* Audit Trails:  
  Maintain proper audit records to meet legal requirements.

**Summary**

The successful development and operation of the airline ticket booking system depends heavily on third-party services (payment, email, hosting) and strict regulatory compliance. Delays or failures in these areas could directly impact system launch and reliability.

1. **REQUIREMENTS**
   1. **Functional Requirements**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Requirement** | **Comments** | **Priority** |
| 1 | Passengers must be able to register and authenticate via email and password. | Includes email verification, secure password rules, and prevention of duplicate accounts. | 1 |
| 2 | Passengers must be able to log in to the system securely. | Includes error handling, lockout after failed attempts, and session redirection. | 1 |
| 3 | Tech Support must be able to log in securely with dedicated credentials. | Enables monitoring and troubleshooting of backend services. | 2 |
| 4 | Admin must be able to log in securely with privileged access. | Access to dashboards, flight management, and report generation. | 1 |
| 5 | Passengers can search for flights by origin, destination, dates, class, etc. | Includes filters like airline, stops, and time range. | 1 |
| 6 | Passengers can book flights and enter passenger information. | Includes contact info, extras (meals, baggage), and booking summary. | 1 |
| 7 | Passengers can select seats during the booking process. | Interactive seating map or grid-based picker with class preference. | 2 |
| 8 | Passengers can make secure payments after booking flights. | Payment methods include credit/debit cards, PayPal, Apple/Google Pay. | 1 |
| 9 | Passengers receive booking confirmations and e-tickets. | Sent via email; includes ticket number and invoice. | 1 |
| 10 | Passengers can reset their password securely. | Email-based reset with secure time-limited tokens. | 1 |
| 11 | Admin can add, update, or delete flights and manage availability. | Includes scheduling, pricing, and capacity editing. | 1 |
| 12 | Admin can cancel flights and notify passengers. | Provides reasons for cancellation and options for refund/rebooking. | 1 |
| 13 | Admin can modify flights and issue compensation. | Affects active bookings; compensation applied automatically or manually. | 1 |
| 14 | Admin can access system analytics dashboards. | Includes traffic data, usage trends, and filtering by date/feature. | 2 |
| 15 | Admin can generate monthly statistical reports. | Aggregates booking trends, reviews, and flight frequency data. | 2 |
| 16 | Admin can access and download financial reports. | Data includes revenue, transactions, refunds; charts and tables available. | 1 |
| 17 | Passengers can enroll in a loyalty program. | Allows earning/redeeming points, upgrading tiers, and viewing balances. | 3 |
| 18 | Passengers receive notifications for bookings and flight updates. | Sent via email/SMS; passengers can manage preferences. | 2 |
| 19 | Passengers can check in online during the valid window. | Generates digital boarding passes with QR codes. | 2 |
| 20 | Tech Support can monitor and maintain system performance. | Diagnoses issues, applies patches, and logs resolutions. | 2 |

* 1. **Non – Functional Requirements**

**3.2.1 Product Requirements**

**3.2.1.1 User Interface Requirements**

* The system must offer a modern, responsive, and visually consistent interface that ensures a seamless experience across all major web browsers (Chrome, Firefox, Safari, Edge) and platforms (Windows, macOS, Linux, iOS, Android). The interface should adapt gracefully to various screen sizes and device resolutions.
* All interface components, including buttons, forms, dropdowns, and navigation panels, must follow universally accepted design principles to promote familiarity and user confidence.
* Search results for flights should be presented in a clean, tabulated format with clearly labeled attributes such as departure time, arrival time, ticket price, and flight class.
* Error messages should be informative, concise, and visually distinct. They must specify the nature of the issue and suggest steps the user can take to resolve it.
* Administrative dashboards should be structured logically with sortable tables, filter options, and exportable reports, enabling quick decision-making and efficient management of operational tasks.

**3.2.1.2 Usability**

* The user interface must be intuitive and user-friendly, allowing even first-time users to perform essential functions such as registration, booking, and payment without requiring technical support or training.
* Tooltips, progress indicators, and inline validation should be integrated throughout the interface to assist users in completing actions correctly and confidently.
* A comprehensive help section must be available, featuring frequently asked questions (FAQs), illustrated walkthroughs, video tutorials, and searchable guidance articles.
* The overall design should minimize user effort by reducing the number of steps required to complete core processes. For instance, the complete flight booking process from search to confirmation should not exceed five user actions.

**3.2.1.3 Efficiency**

***3.2.1.3.1 Performance Requirements***

* The platform must be capable of handling heavy concurrent traffic, supporting a minimum of 500 users simultaneously without noticeable delays or degradation in performance.
* All key operations such as user login, flight searches, and payment processing must complete within two seconds in 95% of cases under normal system loads.
* The response time for critical backend services such as booking confirmation and transaction processing should not exceed three seconds, even under peak load conditions.
* Administrative reporting tools must generate detailed analytics or summaries in under five seconds when processing datasets up to 10,000 entries.

***3.2.1.3.2 Space Requirements***

* The entire system should remain efficient in storage utilization, limiting server-side storage consumption to under 10 GB within the first year of operation.
* System-generated log files must be automatically rotated on a daily basis, compressed for efficient storage, and archived monthly. These logs should be securely deleted after a six-month retention period unless extended by legal or compliance policies.
* Uploaded user documents, transaction records, and downloadable content must be stored in an optimized format and subjected to regular clean-up routines to avoid bloat and redundancy.

**3.2.1.4 Dependability**

***Availability***

* The system must guarantee high availability with a minimum monthly uptime of 99.5%. This includes ensuring access to services such as account management, flight bookings, and support features.
* Any planned downtime or maintenance activities must be communicated to users at least 48 hours in advance and should not exceed a total of four hours per month.

***Reliability***

* The application should be capable of handling intermittent network issues and minor server disruptions without causing service interruptions or data loss. The system should be resilient and capable of self-recovery in such cases.
* Failures must be limited to no more than one occurrence per 1,000 transactions. Each failure must be recorded in detail and trigger appropriate alerts.

***Monitoring***

* Real-time monitoring tools must continuously track the system’s performance, flagging abnormal conditions, and notifying technical support teams via automated alerts.
* Comprehensive logs should be maintained for all login attempts, payment activities, and any system errors. These logs must include timestamps, user IDs, and event specifics to assist in diagnostics and audits.

***Maintenance***

* The software must be designed in a modular fashion to allow quick and safe updates. New features or patches should be deployable independently, without requiring full system redeployment.
* Maintenance activities such as updates, bug fixes, and configuration changes should be executable with minimal user disruption.
* The platform must support rollback capabilities to previous stable versions in case of deployment issues.

**3.2.1.5 Security**

* All passwords must be stored using secure hashing algorithms such as bcrypt or SHA-256, combined with salting techniques to prevent brute-force attacks.
* The entire system must use HTTPS for secure data transmission, and internal APIs must be protected with authentication tokens and TLS encryption.
* Role-based access control (RBAC) must be implemented to define user permissions and restrict access to sensitive data and administrative functions.
* Admin and Tech Support users must be required to use two-factor authentication (2FA) for added protection.
* Payment information must never be stored on the system’s servers. All financial transactions should be handled through PCI-DSS-compliant gateways, and only non-sensitive data should be retained for reference and reporting.

**3.2.2 Organizational Requirements**

**3.2.2.1 Environmental Requirements**

* The platform must be hosted in a containerized infrastructure such as Docker, with compatibility for popular cloud providers including AWS, Azure, and Google Cloud Platform.
* Redundancy must be supported across geographic regions to ensure continuous service in case of local outages or natural disasters.

**3.2.2.2 Operational Requirements**

* Admins must have access to a secure, role-restricted dashboard from which they can view system logs, usage reports, and alert messages.
* The system must allow dynamic configuration of features such as toggling promotions, adjusting booking thresholds, or controlling logging levels without redeployment.
* Operations staff must be able to export log files in human-readable formats such as CSV, JSON, or XML for third-party audits or compliance checks.

**3.2.2.3 Development Requirements**

* The development team must follow Agile methodologies with sprint-based planning and bi-weekly review cycles.
* All project code must be tracked using Git, with branching strategies and pull request reviews enforced as part of continuous integration practices.
* Automated testing tools must be integrated into the development pipeline to ensure code quality, security compliance, and maintainability.

**3.2.3 External Requirements**

**3.2.3.1 Regulatory Requirements**

* The platform must comply fully with the General Data Protection Regulation (GDPR) regarding data retention, user consent, right to erasure, and access controls.
* Privacy policies and terms of service must be made readily accessible and updated transparently when changes occur.

**3.2.3.2 Ethical Requirements**

* The application must not exploit user data for commercial gain without explicit consent. This includes ensuring that no personal or behavioral data is sold or shared with third parties without prior approval.
* Automated features such as refund eligibility, seat assignment, or bonus calculation must operate transparently, and users must be able to review how decisions are made.

**3.2.3.3 Legislative Requirements**

***3.2.3.3.1 Accounting Requirements***

* All transactions must be itemized, time-stamped, and associated with a specific user and session. This applies to both successful and failed attempts.
* Refunds and compensations must include justifications and processing logs to ensure legal accountability.
* Monthly reports generated for accounting purposes must comply with industry-standard formats and be exportable to spreadsheets or financial software.

***3.2.3.3.2 Security Requirements***

* All database changes made by admin users must be recorded with before-and-after data snapshots, along with timestamps and operator IDs.
* The system must implement logging for all sensitive activities including user role updates, system configuration changes, and payment API calls.
* Access logs and audit trails must be securely stored for a minimum of six months and only accessible to users with appropriate clearance.
  1. **Domain Requirements**
* The system must incorporate airline-specific capabilities such as frequent flyer enrollment, point redemption, real-time seat availability tracking, and class-based ticket differentiation.
* Industry-standard practices such as IATA flight codes, baggage rules, and cancellation policies must be integrated into the system’s operational logic.
* Integration with third-party tools such as airport data services or airline API feeds must be supported for up-to-date flight scheduling and disruption alerts.
* The loyalty program must reflect industry-standard models, allowing for dynamic tier upgrades, retroactive point accrual, and partner airline collaboration.

These non-functional requirements ensure the airline booking platform is not only feature-rich but also secure, scalable, and reliable—ensuring a smooth and satisfying experience for all stakeholders involved.

1. **USER SCENARIOS – USE CASES**

## **4.1 Major Functions:**

## **Account Management and Authorization**

* **Registration and Login:**  
  Users can create accounts, securely log in, and recover their password if needed.
* **Profile Management:**  
  Users can manage their personal information, update contact details, and change or add payment methods.

### **Passenger Services and Feedback**

* **Flight Search and Booking:**  
  Passengers can search for flights by various criteria (e.g., destination, date, class), view detailed flight information, and proceed with booking and payment.
* **Booking Management:**  
  Passengers can cancel, modify, or rebook flights if needed. They can also view the status of their booking (confirmed, pending, canceled).
* **Feedback and Ratings:**  
  Passengers can provide feedback about their flight experience and rate their service through the system. This feedback helps improve the service quality.

### **Financial Transaction Management**

* **Revenue Performance Reports:**  
  Finance teams can access comprehensive reports on the system's revenue performance, including charts and graphs showing data by route, booking time, and other relevant metrics.

### **Flight Management by Air Control**

* **Flight Planning and Management:**  
  Authorized personnel can create and modify flight plans. This includes adding or deleting flights, modifying departure/arrival times, aircraft details, and other key flight information.

### **Operator and Report Management by Managers**

* **User Account Oversight:**  
  Managers can oversee user accounts (except admins), approve changes, and ensure proper security and compliance. They use specific authorization codes for these actions.
* **Performance Monitoring and Reporting:**  
  Managers can track system performance, review operator performance, access operational data, and generate reports based on customer feedback, flight frequency, and booking trends.

### **Customer Support by Operators**

* **Customer Support Information:**  
  The system provides passengers with easy access to customer support contact information (e.g., phone, email, live chat).
* **Notifications:**  
  Automated notifications are sent to passengers about booking confirmations, cancellations, or any changes to their flight. These are sent via email or SMS as per the user’s preferences.

## **4.2 Use Cases**

1. **User – Passenger Registration & Authentication**

*The system shall enable users (passengers) to initialize the creation of new accounts.*

|  |  |
| --- | --- |
| **UC Name** | **User – Passenger Registration UC – 101** |
| **Summary** | Enabling passengers to register & create new accounts within the system. |
| **Dependency** | None |
| **Actors** | Primary Actor: User (Passenger)  Secondary Actor: System |
| **Pre-conditions** | 1. The user attempts to sign up using personal information.  2. The user’s unique account identifiers (email) do not match with other users in the system.  3. The user agrees to the terms and conditions of service before proceeding with the sign-up process.  4. The user provides all required information fields (such as name, email, password) during the sign-up attempt. |
| **Description of the Main Sequence** | *Step 1*: The user navigates to the sign-up page on the system.  *Step 2*: The user fills in the required personal information such as name, email address, and password.  *Step 3*: The system validates the entered information to ensure all required fields are filled correctly.  *Step 4:* The system checks if the provided email address is unique and not already associated with an existing account.  *Step 5*: If the email address is unique, the system sends a verification email to the provided address.  *Step 6*: The user receives the verification email and clicks on the verification link to confirm their email address.  *Step 7:* Upon email verification, the system creates a new account for the user.  *Step 8:* The user receives a confirmation message indicating successful account creation.  *Step 9:* The user can now log in to the system using their email address and password. |
| **Description of the Alternative Sequence** | *Alternative sequence 1:* Email already in use: If the email is already registered, the system notifies the user and prevents registration.  *Alternative sequence 2:* Missing required fields: If any required field is left blank, the system prompts the user to fill it.  *Alternative sequence 3:* Verification link expired or invalid: If the link is not used within a certain time, the user must request a new one. |
| **Non – functional Requirements** | *Security:* The security requirement for the sign-up process ensures the protection of user accounts and sensitive information from unauthorized access and potential breaches.  *Performance:* The authentication process is expected to conclude within a specified duration of seconds.  *Scalability:* The authentication database system should be capable of handling a large number of accounts. |
| **Post – conditions** | A new user account is created and stored in the system.  The user is authenticated and redirected to the dashboard (or login page).  User’s email is marked as verified in the system. |

*Login-The system shall allow users and passengers to securely authenticate themselves and gain access to their accounts.*

|  |  |
| --- | --- |
| **UC Name** | **User Account Log – in  UC – 102** |
| **Summary** | Enabling passengers to securely access their accounts within the system. |
| **Dependency** | User Registration (UC-101) |
| **Actors** | Primary Actor: User/Passenger  Secondary Actor: System |
| **Pre-conditions** | 1.The user (passenger) has already created an account with the system or has been given one by the Administrator of the Airline.  2.The user possesses valid credentials: registered email address &password.  3.The user has agreed to the terms and conditions before attempting log in.  4.The system is operational and functional in order to allow login. |
| **Description of the Main Sequence** | *Step 1:* The user enters their credentials to log in to the system.  *Step 2:* The system verifies the provided credentials against the stored user data.  *Step 3:* If the credentials match an existing user account, the system grants access to the user. |
| **Description of the Alternative Sequence** | *Alternative Sequence 1:* If the credentials do not match or are invalid, the system denies access and prompts the user to try again or reset their password.  *Alternative Sequence 2:* If any errors occur during the log-in process, such as invalid information or technical issues, the system provides appropriate error messages and prompts the user to correct the issues and try again. |
| **Non – functional Requirements** | *Security:* The security requirement for the login process ensures the protection of user accounts and sensitive information from unauthorized access and potential breaches.  *Performance:* The authentication process is expected to conclude within a specified duration of seconds.  *Scalability:* The authentication database system should be capable of handling a large number of accounts. |
| **Post – conditions** | 1. If the user's credentials are valid, they gain access to their account.  2. Upon successful login, the system may redirect the user to their account dashboard.  3. Invalid credentials result in access denial with error messages.  4. After multiple failed login attempts (maybe 3 or 4) , the system may lock the account for security. |

*System shall allow passengers/users to reset their account password.*

|  |  |
| --- | --- |
| **UC Name** | **User Password Reset UC – 103** |
| **Summary** | Enabling passengers to securely reset their accounts within the system. |
| **Dependency** | User Account Creation (UC-101) |
| **Actors** | Primary Actor: User  Secondary Actor: System |
| **Pre-conditions** | The user has a registered account in the system.  The user initiates reset process from the login or help screen.  The user provides a valid, registered email address.  The system must be able to send emails to the user's address. |
| **Description of the Main Sequence** | *Step 1:* The user clicks the "Forgot Password" button. *Step 2:* The system verifies that the user email exists in the user database. *Step 3:* The system sends a password reset email containing a secure, time-limited link. *Step 4:* The user receives the email and clicks the password reset link. *Step 5:* The system prompts the user to enter and confirm a new password. *Step 6:* The system validates the new password (e.g., complexity rules, matching confirmation). *Step 7*: The system updates the user's password in the database. *Step 8:* The user receives confirmation that their password has been successfully reset. *Step 9:* The user can now log in using the new password. |
| **Description of the Alternative Sequence** | Alternative Sequence: Invalid Email:  *Step 1:* The user enters a new password and submits the form.  *Step 2:* The system checks the password against the defined policy.  *Step 3:* The system detects that the password does not comply with the rules.  *Step 4:* The system rejects the password and displays a descriptive validation message.  *Step 5:* The user remains on the same form and is prompted to re-enter a stronger password.  *Step 6:* If the new password still does not meet requirements, the system repeats Steps A2–A5.  *Step 7:* Once the user enters a password that meets the criteria, the system proceeds to the next step in the Main Sequence (Step 4). |
| **Non – functional Requirements** | *Security:* The security requirement for the reset process ensures the protection of user accounts and information from unauthorized access and potential breaches.  *Usability:* Maintain an intuitive user interface. |
| **Post – conditions** | The user has a registered account in the system.  The user initiates the password reset process from the login or help screen.  The user provides a valid, registered email address.  The system must be able to send emails to the user's address. |

1. **Flight Search & Filtering:**

*The system shall enable users (Passengers) to initiate the creation of new accounts.*

|  |  |
| --- | --- |
| **UC Name** | **Filter Flights  UC – 201** |
| **Summary** | The system must enable users to search for different flights efficiently. This feature allows users to specify their travel preferences and find relevant flight options based on criteria such as origin, destination, departure date, price, and class. By providing robust flight search functionality, the system enhances user experience and facilitates seamless flight booking. |
| **Dependency** | User Registration (UC-101) |
| **Actors** | Primary Actor: Passengers |
| **Pre-conditions** | Before utilizing the flight search functionality, users must be authenticated and logged into their account within the airline software, ensuring that only authorized users can access and utilize the search feature. |
| **Description of the Main Sequence** | *Step 1:* User Input: Users input their flight preferences, such as departure city, destination, travel dates, price, and class, into the search interface.  *Step 2:* Query Submission: Upon inputting preferences, users submit their search query by clicking a search button or similar action.  *Step 3:* Search Processing: The system processes the search query, analyzing the user's input criteria to retrieve relevant flight options from the database.  *Step 4:* Flight Retrieval: Based on the search criteria, the system retrieves available flight options that match the user's preferences, including flight schedules, fares, and availability.  *Step 5:* Display Results: The system displays the retrieved flight options in a clear and organized manner, presenting essential details such as departure times, arrival times, airlines, and ticket prices.  *Step 6:* Refinement Options: Users may have options to refine their search results further, such as filtering by airline, price range, departure time, or number of stops.  *Step 7:* Selection: Users review the displayed flight options and select the one that best fits their preferences and requirements.  *Step 8:* View Details: Upon selecting a flight, users may have the option to view additional details, such as seat availability, aircraft type, in-flight amenities, and fare conditions.  *Step 9:* Return to Search: Users have the option to return to the flight search interface to perform additional searches or explore alternative flight options if needed. |
| **Description of the Alternative Sequence** | *Step 1:* User Input: Users input their flight preferences, such as departure city, destination, travel dates, and class, into the search interface.  *Step 2:* Query Submission  Upon inputting preferences, users submit their search query by clicking a search button or similar action.  *Step 3:* Search Processing  The system processes the search query, analyzing the user's input criteria to retrieve relevant flight options from the database.  *Step 4:* Flight Retrieval  The system checks the database for available flight options that match the user's preferences.  *Step 5*: No Matching Flights Found  If there are no flights that match the user's input criteria, the system informs the user that no matching flights were found.  *Step 6:* Error Handling: The system may provide suggestions to the user, such as adjusting the search criteria, selecting alternative travel dates, or considering nearby airports.  *Step 7:* Return to Search: Users have the option to return to the flight search interface to modify their search criteria and perform a new search. |
| **Non – functional Requirements** | *Performance:* Ensure fast response times.  *Usability:* Maintain an intuitive user interface.  *Accessibility:* Comply with accessibility standards.  *Reliability:* Provide accurate and reliable search results.  *Scalability:* Handle increasing user load without performance degradation.  *Security:* Protect user data during search queries. |
| **Post – conditions** | After selecting a flight, users are seamlessly guided through the booking process, where they can confirm their flight selection, provide necessary passenger information, and complete the reservation. Upon successful booking, users receive a confirmation of their flight reservation along with relevant booking details. |

1. **Flight Booking and Payment**

*The system shall allow the user to book a selected flight.*

|  |  |
| --- | --- |
| **UC Name** | **Book Flight  UC – 301** |
| **Summary** | The system must enable users to book a selected flight seamlessly. This feature allows users to confirm their flight selection, provide passenger details, and complete the reservation process efficiently within the airline software. By providing robust flight booking functionality, the system enhances user experience and facilitates bookings |
| **Dependency** | User Registration & Login (UC-101&102)  Filter Flights (UC-201) |
| **Actors** | Primary Actor: Passenger |
| **Pre-conditions** | Passenger has searched the flight and selected to see the details for that flight |
| **Description of the Main Sequence** | *Step 1:* Flight Selection: After searching for and selecting a desired flight, users proceed to book the chosen flight.  *Step 2:* Flight Details Review: Users review the details of the selected flight, including departure and arrival times, fares, and other relevant information.  *Step 3:* Passenger Information: Users provide necessary passenger details such as names, contact information, and any special requirements.  *Step 4:* Seat Selection (if applicable): If seat selection is available, users may choose their preferred seats or seating options for the flight.  *Step 5:* Additional Services (if applicable): Users may have the option to select additional services such as baggage allowance, meal preferences, or seat upgrades.  *Step 6:* Booking: Users finalize the flight booking |
| **Description of the Alternative Sequence** | Flight Unavailability  *Step 1:* If the selected flight becomes unavailable due to real-time updates, the system notifies the user.  *Step 2:* User is taken back to the Flight Filtering tab where he/she searches for similar alternative flights. |
| **Non – functional Requirements** | *Performance*: Ensure fast response times during the booking process.  *Usability*: Maintain an intuitive and user-friendly interface for seamless booking.  *Accessibility*: Comply with accessibility standards to accommodate users with disabilities.  *Reliability*: Provide reliable booking functionality with minimal downtime.  *Scalability*: Handle concurrent booking requests from multiple users without performance degradation.  *Error Handling*: Effectively handle errors and edge cases during booking to ensure a smooth user experience.  *Availability*: Ensure the booking system is available 24/7 to accommodate users. |
| **Post – conditions** | After successfully completing the booking process, users receive a summary of their booking and proceed to the payment step |

*The system shall allow the user to pay for a booked flight.*

|  |  |
| --- | --- |
| **UC Name** | **Flight Payment Processing  UC – 302** |
| **Summary** | The system must enable users to book a selected flight seamlessly. This feature allows users to confirm their flight selection, provide passenger details, and complete the reservation process efficiently within the airline software. By providing robust flight booking functionality, the system enhances user experience and facilitates hassle-free flight reservations. |
| **Dependency** | Secure Payment Transactions |
| **Actors** | Primary Actor: Passenger |
| **Pre-conditions** | Passenger has booked the particular flight.  (Optionally- has redeemed loyalty points) |
| **Description of the Main Sequence** | *Step 1:* User selects the preferred payment method (credit/debit card). *Step 2*: User enters payment details (e.g., card number, expiry date, CVV, billing address). *Step 3:* System validates the payment details and checks for sufficient funds. *Step 4:* System processes the payment through the payment gateway. *Step 5*: If payment is successful, the system generates a booking confirmation and sends a receipt via email/SMS. *Step 6*: User is redirected to the confirmation page with booking details and generated ticket from those details. |
| **Description of the Alternative Sequence** | Payment Failure Due to Insufficient Funds  *Step 1*: User selects the preferred payment method and enters payment details. *Step 2:* System attempts to process the payment but detects insufficient funds in the user’s account. *Step 3*: System displays an error message informing the user of the failed transaction. *Step 4:* User is prompted to either: Try another payment method. Pay later.  *Step 5*: If the user selects a different payment method and successfully completes the payment, they return to the main sequence at Step 5 (confirmation). *Step 6:* If the user does not complete the payment, the system cancels the booking attempt after a specified time (e.g., 15 minutes). |
| **Non – functional Requirements** | *Performance*: Ensure fast response times during the payment process.  *Usability*: Maintain an intuitive and user-friendly interface.  *Accessibility*: Comply with accessibility standards to accommodate users with disabilities.  *Reliability*: Provide reliable payment functionality.  *Scalability*: Handle concurrent requests from multiple users without performance degradation.  *Security*: Safeguard user payment and personal information.  *Error Handling*: Effectively handle errors and edge cases.  *Availability*: Ensure the booking system is available 24/7 to accommodate users from different time zones. |
| **Post – conditions** | After successfully completing the booking process, users receive a confirmation of their flight reservation along with relevant booking details. The system generates electronic tickets or booking references, which users can use for check-in and boarding. Optionally, users may receive an email confirmation of their booking for their records. |

1. **Loyalty Program**

*User enrolls in loyalty program.*

|  |  |
| --- | --- |
| **UC Name** | **Loyalty Program  UC – 401** |
| **Summary** | Allows users to enroll in a frequent flyer program and earn points for redemption. |
| **Dependency** | User Registration & Login (UC 101&102) Optionally (Loyalty Program UC 401) |
| **Actors** | Primary actor: User/Passenger  Secondary actors: System |
| **Pre-conditions** | 1. The user must have a valid account.  2. The user must agree to the program's terms. |
| **Description of the Main Sequence** | *Step 1:* User accesses the frequent flyer program enrollment page.  *Step 2:* User provides necessary information and submits.  *Step 3:* System validates the information and enrolls the user.  *Step 4:* Points are earned for eligible purchases or flights.  *Step 5*: User redeems points for discounts or benefits. |
| **Description of the Alternative Sequence** | *Step 1:* If the user fails to provide valid information, the system denies enrollment and displays an error message  *Step 2*: User retries with correct information. |
| **Non – functional Requirements** | *Performance*: Ensure fast response of the program  *Usability*: Maintain an intuitive and user-friendly interface.  *Accessibility*: Comply with accessibility standards to accommodate users with disabilities.  *Reliability*: Provide reliable payment functionality.  *Scalability*: Handle concurrent requests from multiple users without performance degradation.  *Security*: Safeguard user personal information.  *Availability*: Ensure the booking system is available 24/7 to accommodate users from different time zones. |
| **Post – conditions** | The user is enrolled in the frequent flyer program and can earn/redeem points successfully. |

1. **Airline Admin Panel**

*Adds new flights.*

|  |  |
| --- | --- |
| **UC Name** | **Add flight  UC – 501** |
| **Summary** | This use case allows administrators to add new flight details, including route, schedule, and capacity. |
| **Dependency** | None |
| **Actors** | Primary actor: Administrator  Secondary actors: System (flight database platform) |
| **Pre-conditions** | Necessary permissions must be assigned to admin. |
| **Description of the Main Sequence** | *Step 1*: Administrator accesses the flight management interface by logging in using their credentials.  *Step 2*: Administrator navigates to the "Add Flight" section and enters new flight details such as flight number, route, schedule, capacity, and pricing.  *Step 3*: System validates the entered flight details, checking for missing fields or invalid formats, and ensures the flight number is unique.  *Step 4*: System saves the new flight details in the database once validation is successful.  *Step 5*: Administrator receives a confirmation message, and the new flight is made available on user-facing platforms for booking. |
| **Description of the Alternative Sequence** | *Step 1*: Administrator provides incorrect or incomplete flight details, prompting the system to display an error message.  *Step 2*: Administrator corrects the information and resubmits the details.  *Step 3*: If the system fails during the saving process (e.g., due to a network issue), the system notifies the administrator to retry or seek support. |
| **Non – functional Requirements** | *Performance*: System must handle flight additions within 5 seconds.  Usability & Accessibility-Interface must be intuitive and accesible.  *Security*: Data must be stored securely to prevent unauthorized access. |
| **Post – conditions** | New flight is added and made available for users to view and book. |

*Admin authorizes Flight Cancelation. (Optional)*

|  |  |
| --- | --- |
| **UC Name** | **Cancel Flight  UC – 502** |
| **Summary** | This use case allows administrators to cancel flights affected by natural disasters and notify passengers. |
| **Dependency** | UC-501 |
| **Actors** | Primary actor:  Administrator  Secondary actors: System |
| **Pre-conditions** | 1. Natural disaster impacts flight safety or viability.  2. Administrator has proper access rights. |
| **Description of the Main Sequence** | *Step 1:* Administrator navigates to the "Cancel Flight" section and searches for affected flight.  *Step 2:* Administrator selects the flight(s) to cancel and specifies the reason (e.g., natural disaster).  *Step 3:* System prompts for confirmation with flight details and cancellation reason.  *Step 4:* Upon confirmation, the system cancels the flight(s) and updates user-facing platforms  *Step 5*: System notifies affected passengers about the cancellation and provides refund or rebooking options.  *Step 6:* System logs the cancellation for auditing and compliance purposes. |
| **Description of the Alternative Sequence** | *Step 1:* Administrator cannot find the flight in the database, prompting the system to display an error or suggest similar matches.  *Step 2*: Admin chooses the correct flight and process continues as in the main sequence |
| **Non – functional Requirements** | Performance: System must handle flight deletions quickly (within 5 seconds).  Usability & Accessibility: Interface must be intuitive and accessible.  Security-Data must be stored and then deleted to securely to prevent unauthorized access. |
| **Post – conditions** | Flight is successfully canceled, passengers are notified, and the system reflects updated flight schedules. |

*Airline Admin modifies existing bookings.*

|  |  |
| --- | --- |
| **UC Name** | **Modify Flight and Issue Compensation  UC – 503** |
| **Summary** | This use case allows administrators to modify existing flight details and compensate affected clients with active bookings. |
| **Dependency** | UC-101,102, 201, 301, 501 |
| **Actors** | Primary actor: Administrator  Secondary actors: System |
| **Pre-conditions** | The flight must already exist in the system.  The administrator must have modification rights. |
| **Description of the Main Sequence** | *Step 1*: Administrator accesses the flight management interface.  *Step 2*: Administrator selects the flight to modify by searching the flight database.  *Step 3*: Administrator edits the required flight details (e.g., schedule, route, or aircraft type).  *Step 4*: System validates the changes and checks for conflicts  *Step 5*: System applies the modifications and updates related systems  *Step 6*: System identifies all affected clients, if any, with active bookings and allocates the compensation fee based on the policy.  *Step 7*: System notifies affected clients about the modification and compensation via email or SMS.  *Step 8*: System processes the compensation fee and credits it to the affected clients. |
| **Description of the Alternative Sequence** | *Step 1*: If the flight cannot be located in the database, the system displays an error message or suggests similar flights.  *Step 2*: Administrator abandons the process or searches for another flight.  *Step 3*: If compensation processing fails (e.g., payment gateway issue), the system logs the failure and notifies the administrator for follow-up. |
| **Non – functional Requirements** | *Performance*: System must handle flight deletions quickly (within 5 seconds).  *Usability & Accessibility*: Interface must be intuitive and accessible.  *Security*: Data must be stored and then deleted to securely to prevent unauthorized access. |
| **Post – conditions** | Flight details are successfully updated, affected clients are compensated, and the system reflects the modifications. |

*The system shall grant admins access to comprehensive analytics, aiding informed decision-making.*

|  |  |
| --- | --- |
| **UC Name** | **Analytics Access  UC – 504** |
| **Summary** | This use case allows administrators to access detailed analytics within the system, facilitating informed decision-making processes. |
| **Dependency** | None |
| **Actors** | Primary Actor: Airline Administrator  Secondary Actor: System |
| **Pre-conditions** | The system must be operational and accessible.  The administrator must be authenticated and logged into the system.  Detailed analytics must be generated and available within the system.  The admin must have the necessary permissions to access analytics. |
| **Description of the Main Sequence** | *Step 1*: The admin navigates to the analytics section within the system.  *Step 2*: The system presents a list of available analytics options.  *Step 3*: The admin selects the desired analytics to view.  *Step 4*: The system generates and displays the selected analytics data.  *Step 5*: The admin analyzes it to gain insights and make informed decisions.  *Step 6*: The use case concludes, returning the admin to the main interface or allowing them to continue accessing additional analytics as needed. |
| **Description of the Alternative Sequence** | *Step 1*: In case of system errors or unavailability, the admin receives a notification and is prompted to retry accessing analytics later.  *Step 2*: If the selected analytics data is not available or cannot be generated, the system displays an error message and prompts the admin to choose another option.  *Step 3*: Should there be discrepancies or inconsistencies in the analytics data, the system provides additional explanations or guidance.  *Step 4*: The use case concludes, either returning the administrator to the main interface or allowing them to continue accessing and analytics. |
| **Non – functional Requirements** | *Performance*: Ensure prompt response and scalability of the performance management system.  *Security*: Restrict access to authorized personnel, encrypt data, and maintain audit trails for accountability.  *Reliability*: Maintain high availability and perform regular backups for data recovery.  *Usability*: Provide an intuitive interface and allow customization options to enhance user experience.  *Compliance*: Adhere to data privacy regulations and retention policies to ensure legal compliance. |
| **Post – conditions** | The admin has successfully accessed and utilized the desired analytics, enabling them to make informed decisions based on the presented information. |

*The system shall produce monthly statistical reports covering the entire flight booking process, incorporating insights from client reviews and flight frequency data.*

|  |  |
| --- | --- |
| **UC Name** | **Generate Monthly Statistical Report  UC – 505** |
| **Summary** | This use case entails the system's ability to produce monthly statistical reports regarding the maintenance of the flight booking process. These reports encompass the entire booking process, from initiation to completion, and provide valuable insights derived from client reviews and flight frequency data. |
| **Dependency** | None |
| **Actors** | Primary Actor: Airline Admin  Secondary Actor: System |
| **Pre-conditions** | The system is operational and accessible.  The administrator must be authenticated and logged into the system.  Sufficient data related to the flight booking process, including client reviews and flight frequency data, is available for analysis.  The Administrator has access to the statistical reports and intends to utilize the data, including flight frequency information, for analysis and decision-making purposes  There are no ongoing system maintenance activities or technical issues that hinder report generation.  The designated time period for generating monthly statistical reports has commenced (e.g., beginning of a new month). |
| **Description of the Main Sequence** | *Step 1*: The Admin accesses the system and selects the option to generate a monthly statistical report.  *Step 2*: The system collects data on the flight booking process, including client reviews and flight frequency.  *Step 3*: Using the collected data, the system generates a comprehensive report.  *Step 4*: The Admin reviews and finalizes the report.  *Step 5*: The system compiles and presents the report in a suitable format. |
| **Description of the Alternative Sequence** | *Step 1*: The Admin selects the option to generate a monthly statistical report.  *Step 2*: The system encounters an error while collecting data on the flight booking process.  *Step 3:* The system prompts the Admin with an error message indicating the issue.  *Step 4*: The Admin attempts to troubleshoot the error by reinitiating the data collection process.  *Step 5*: The system successfully collects the necessary data and proceeds with generating the report.  *Step 6*: The Admin reviews and finalizes the report as usual.  *Step 7*: The system compiles and presents the report in a suitable format. |
| **Non – functional Requirements** | *Performance*: Fast response time, scalable for growth.  *Reliability*: High uptime, quick recovery from failures.  *Security*: Secure authentication, encrypted data.  *Maintainability*: Modular design, comprehensive documentation. |
| **Post – conditions** | The monthly statistical report for the flight booking process has been generated and is available for review by the Admin. |

*The system shall allow administrators to access financial reports.*

|  |  |
| --- | --- |
| **UC Name** | **Financial Reports  UC – 506** |
| **Summary** | This use case involves administrators accessing financial reports |
| **Dependency** | None |
| **Actors** | Primary Actor: Administrator  Secondary Actor: System |
| **Pre-conditions** | The system must be operational and accessible.  The administrator must be authenticated and logged into the system. |
| **Description of the Main Sequence** | *Step 1*: Administrators are directed to the financial reports interface.  *Step 2*: They request the needed financial data  *Step 3*: The system retrieves the requested financial data and generates the report.  *Step 4*: The system presents the report in a table/chart format.  *Step 5*: Administrators analyze the presented data to gain insights in financial health. |
| **Description of the Alternative Sequence** | *Step 1:* If the system encounters an error and is unable to retrieve the required data, it displays an error message and administrators notify of the issue. |
| **Non – functional Requirements** | *Security*: Access to financial reports should be restricted to authorized shareholders to maintain data confidentiality.  *Usability*: The system interface for accessing financial reports should be intuitive and easy to navigate for shareholders. |
| **Post – conditions** | Administrators have successfully accessed financial reports |

1. **Tech Support / Maintenance**

*Tech Support ensuring Software functionality.*

|  |  |
| --- | --- |
| **UC Name** | **Software Functionality**  **UC – 601** |
| **Summary** | This use case enables the tech support team to identify, troubleshoot, and resolve software issues to ensure smooth operation. |
| **Dependency** | None |
| **Actors** | Primary actor: Tech Support  Secondary actors: System |
| **Pre-conditions** | 1. Software must be deployed and operational.  2. Tech support team must have appropriate access credentials and tools. |
| **Description of the Main Sequence** | *Step 1*: Tech support receives an issue report or identifies a potential problem through monitoring tools.  *Step 2*: Tech support analyzes the issue using diagnostic tools and software logs.  *Step 3*: Root cause is identified, and a solution is developed (e.g., patch, configuration update, or user guidance).  *Step 4*: Solution is applied to the software, and the system is tested to ensure the issue is resolved.  *Step 5*: End users or administrators are notified of the resolution and provided with any necessary follow-up instructions. |
| **Description of the Alternative Sequence** | *Step 1*: If the issue cannot be resolved immediately, tech support escalates it to the development or engineering team.  *Step 2*: Temporary workarounds are provided to minimize user impact while the issue is being addressed.  *Step 3*: If the diagnostic tools fail or provide inconclusive results, the team employs manual investigation methods or consults with external experts. |
| **Non – functional Requirements** | Issues must be identified and acknowledged within 5 minutes of detection or reporting. Resolutions must be implemented with minimal downtime for end users. All actions must be logged for auditing and future reference. |
| **Post – conditions** | This use case enables the tech support team to identify, troubleshoot, and resolve software issues to ensure smooth operation. |

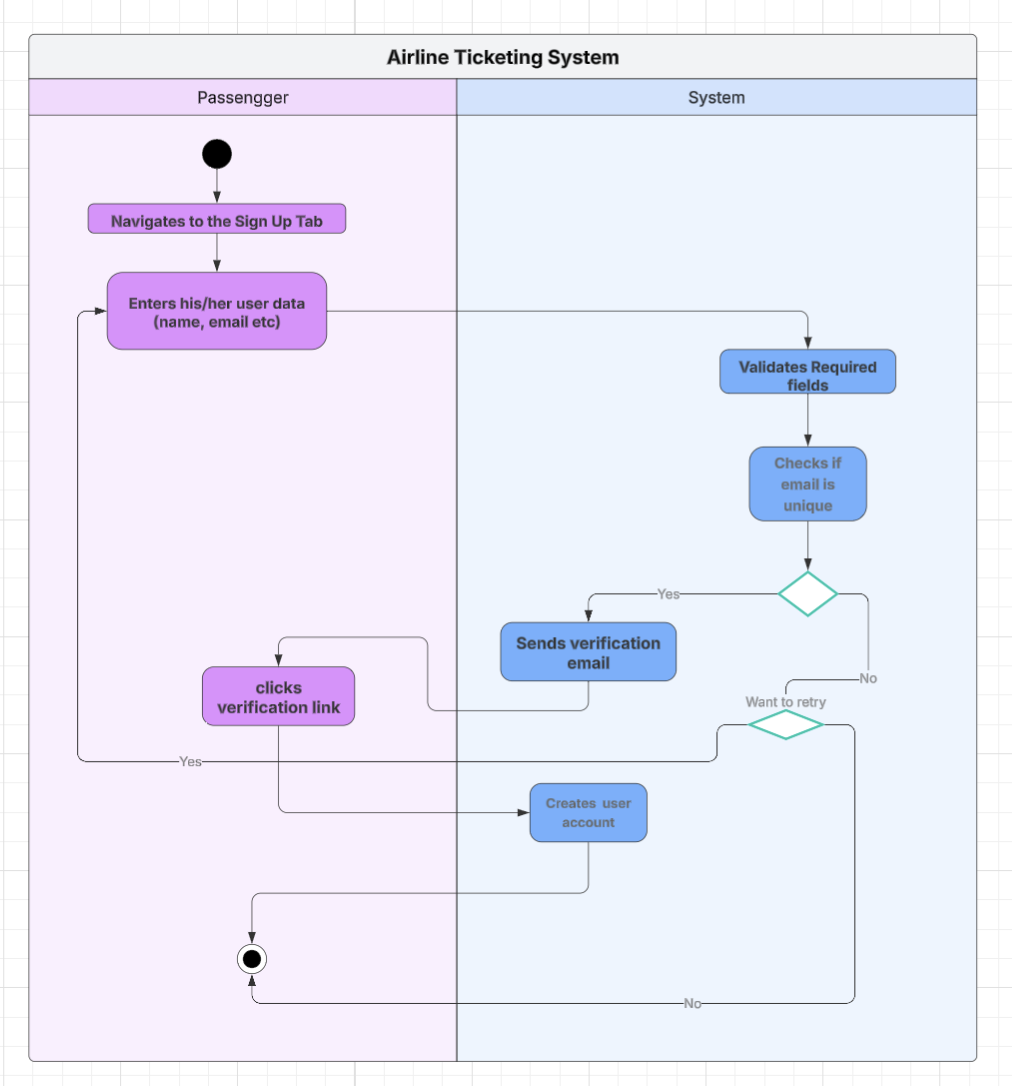
1. **DIAGRAMS**
   1. **Use Case Diagram**

A diagram of a flight ticketing system

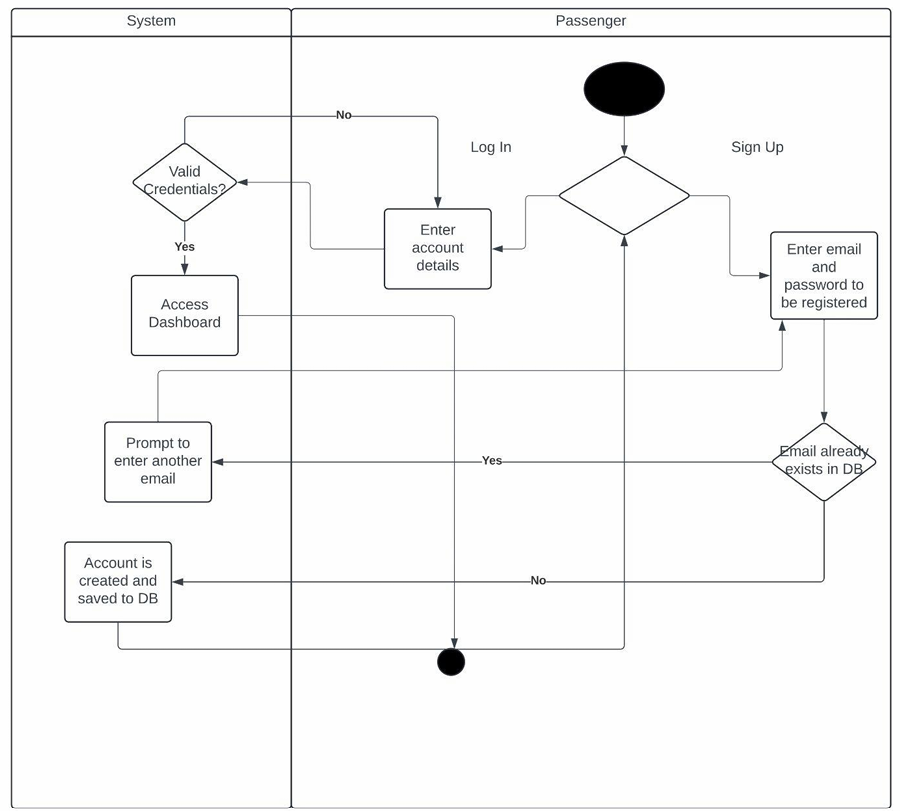
AI-generated content may be incorrect.

* 1. **Activity Diagram**

**UC – 101: User – Passenger Registration**

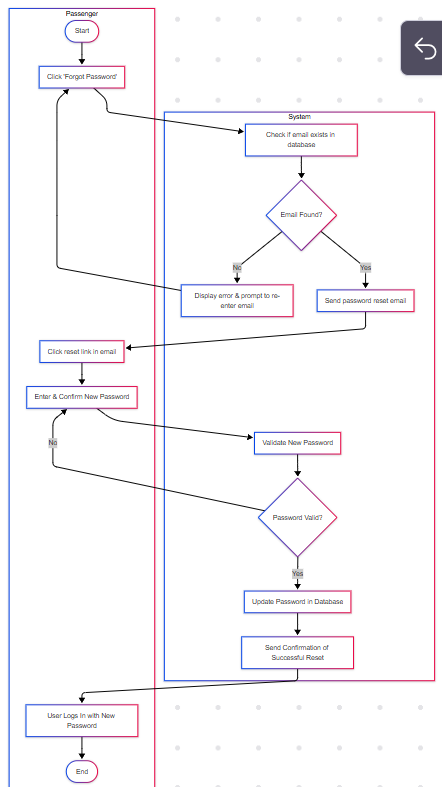


**UC – 102: User Account Log – in**

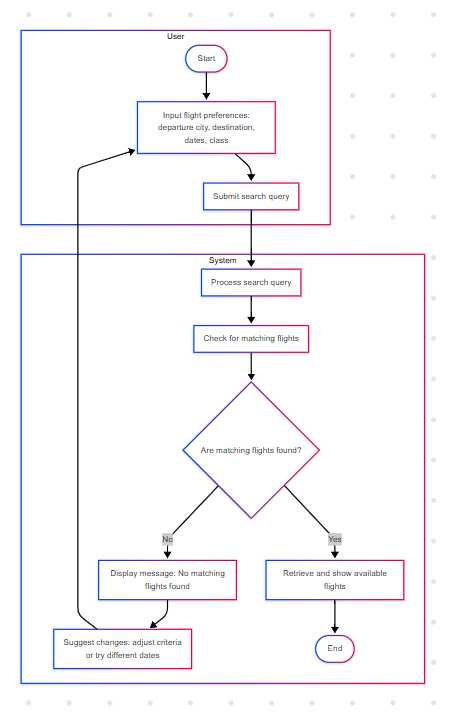


**//LOG IN SIGN UP**

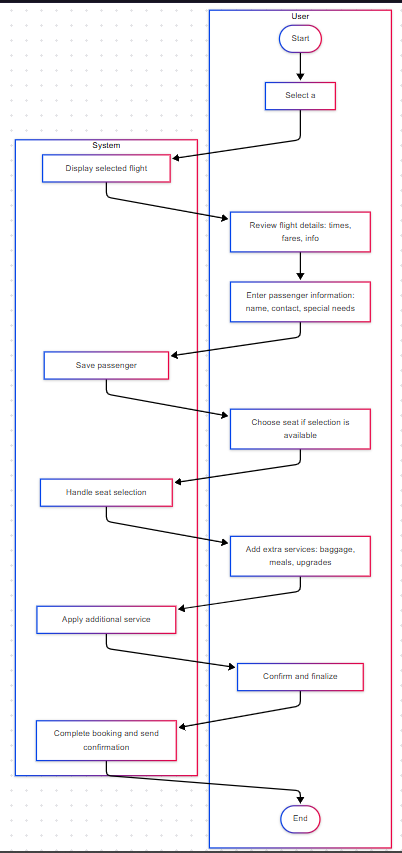
**?**

**UC – 103: User Password Reset**

**UC – 201: Filter Flights**



**UC – 301: Book Flight**



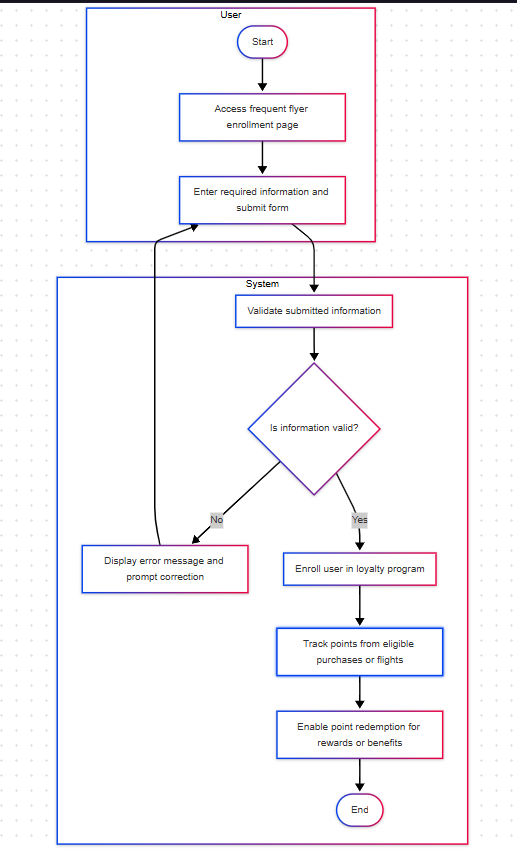
**UC – 302: Flight Payment Processing**

A diagram of a system

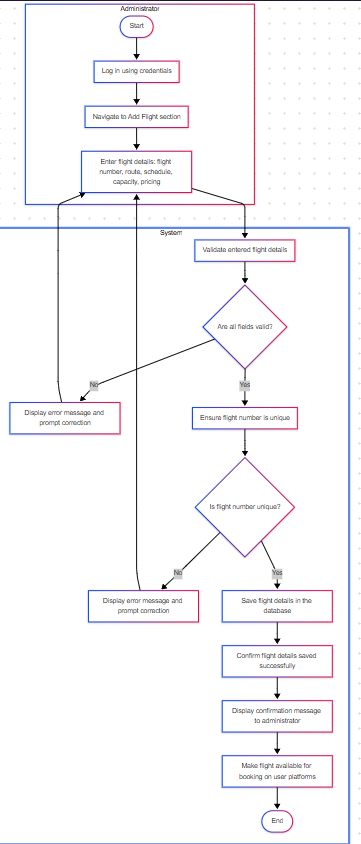
AI-generated content may be incorrect.

**UC – 401: Loyalty Program**

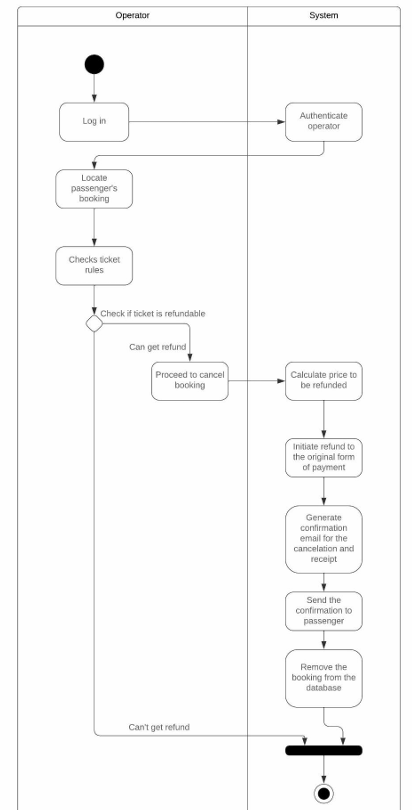
**????????????Manging booked flights at doc week 6??**



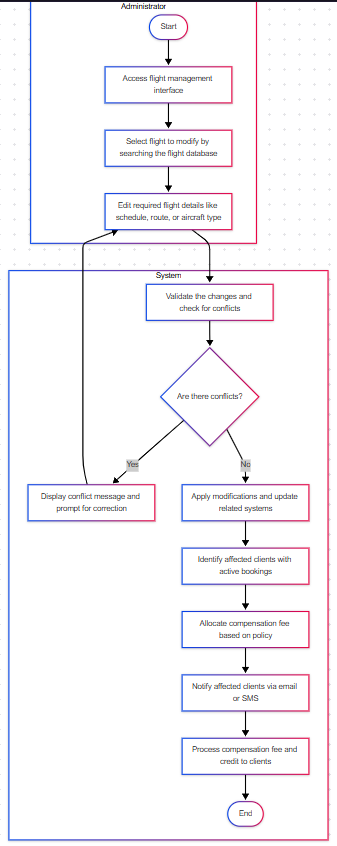
**UC – 501: Add Flight**



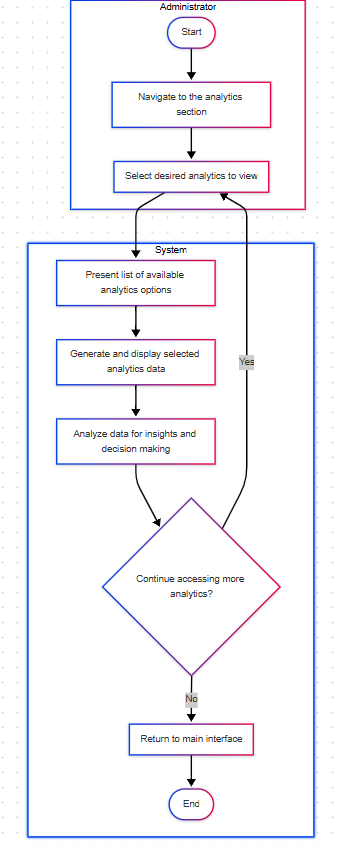
**UC – 502: Cancel Flight**



**UC – 503: Modify Flight and Issue Compensation**



**UC – 504: Analytics Analysis**



**UC – 505: Generate Monthly Statistical Report**

A diagram of a data flow

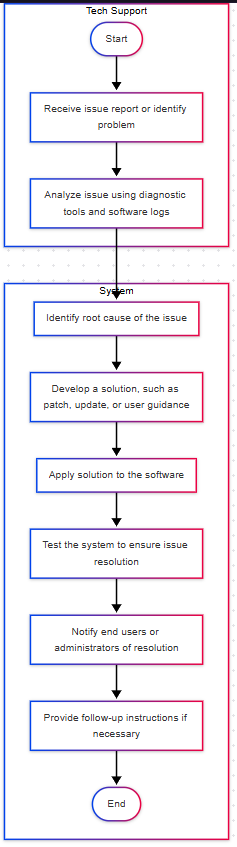
AI-generated content may be incorrect.

**UC – 506: Financial Report**

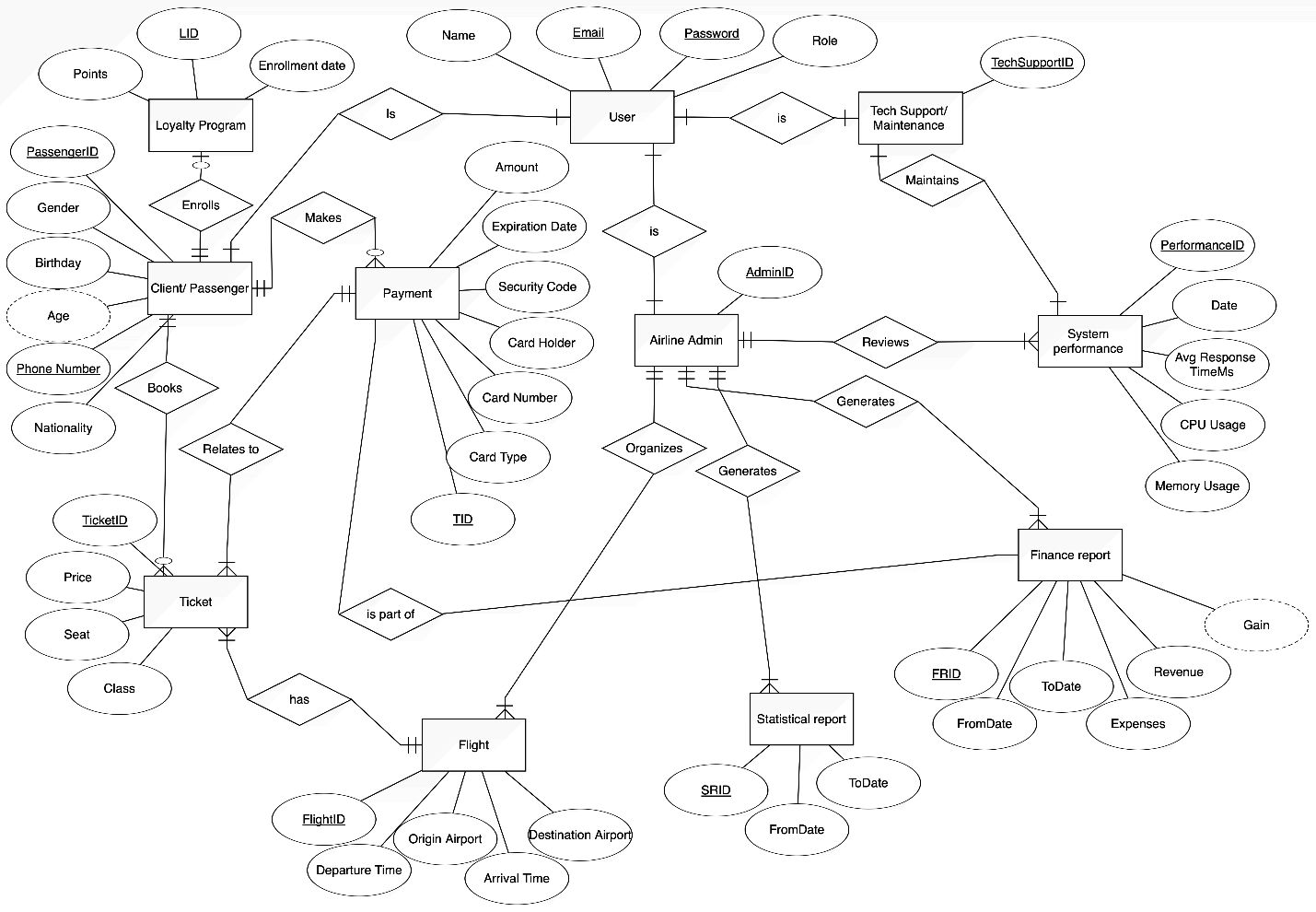
A screenshot of a diagram

AI-generated content may be incorrect.

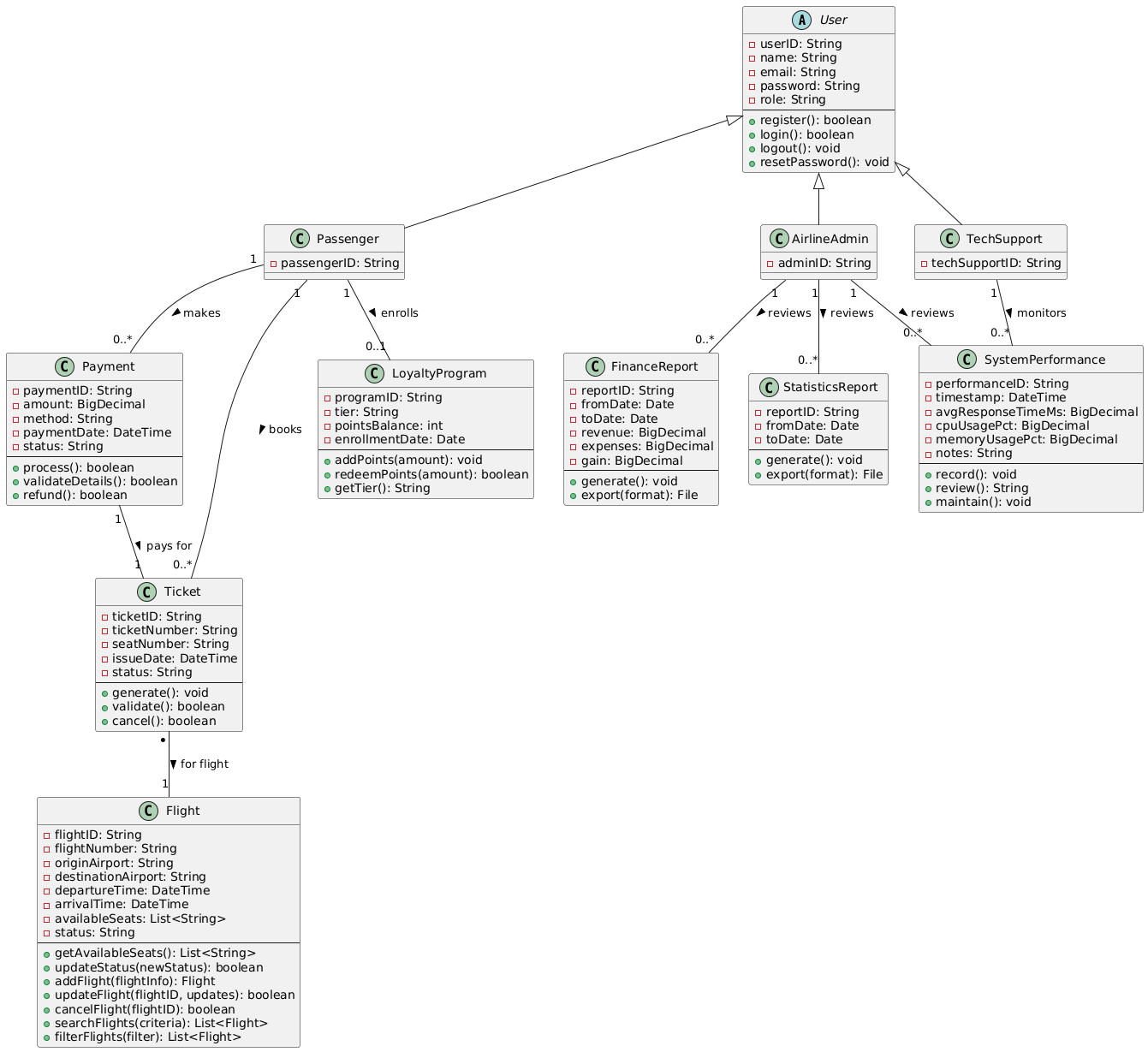
**UC – 601: Software Functionality**



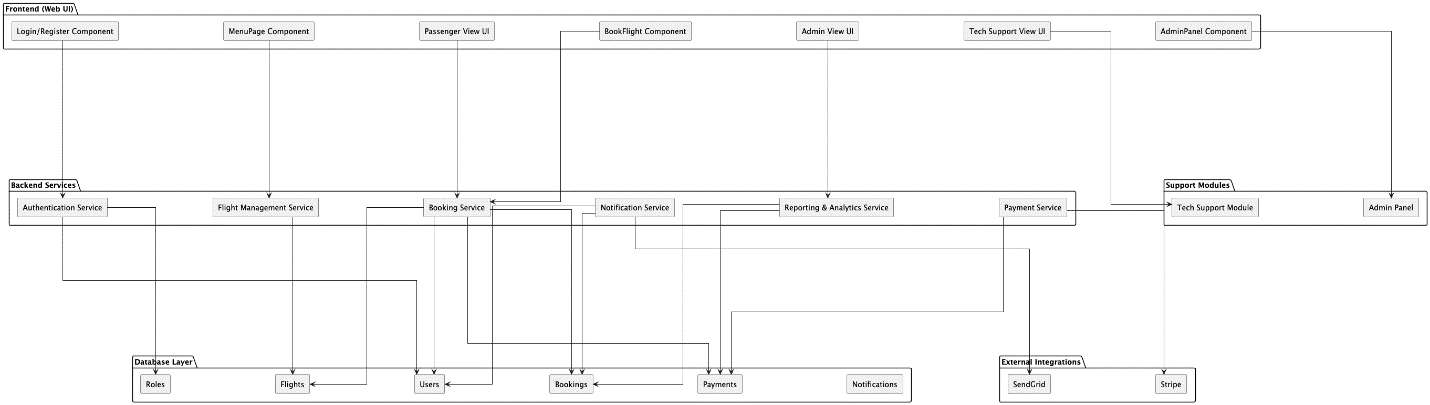
* 1. **ERD Diagram**



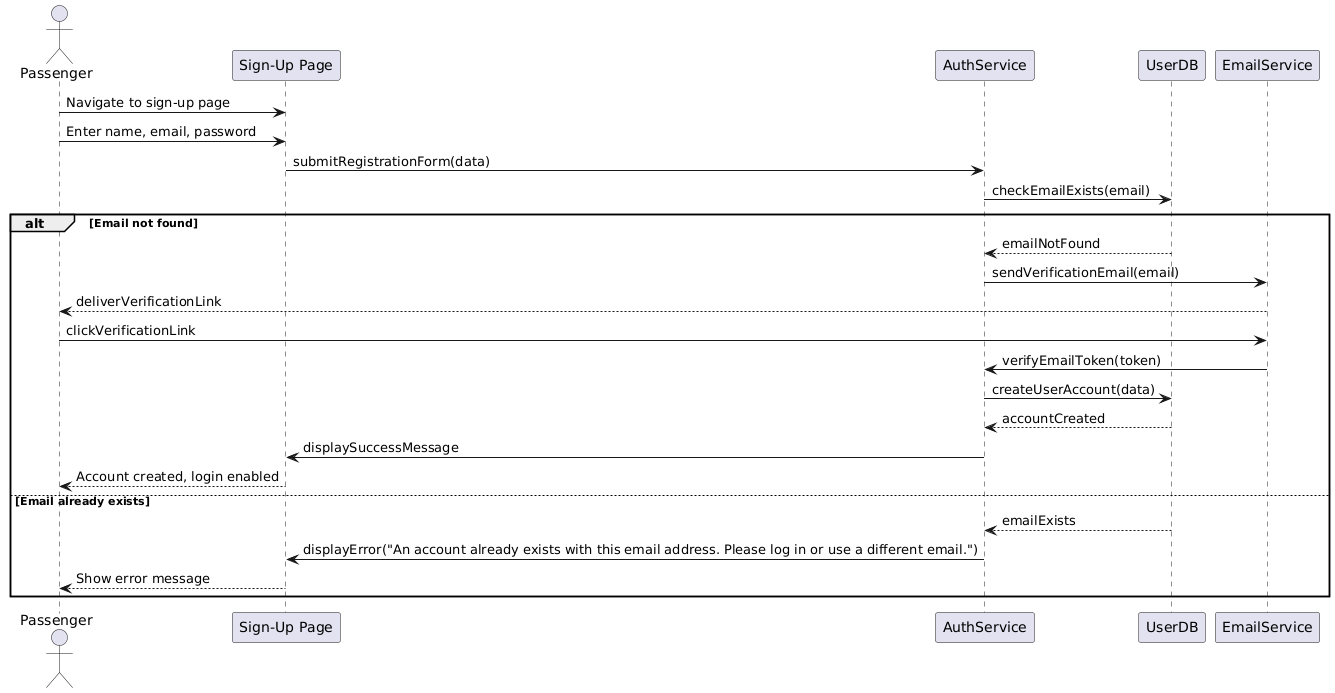
* 1. **Class Diagram**



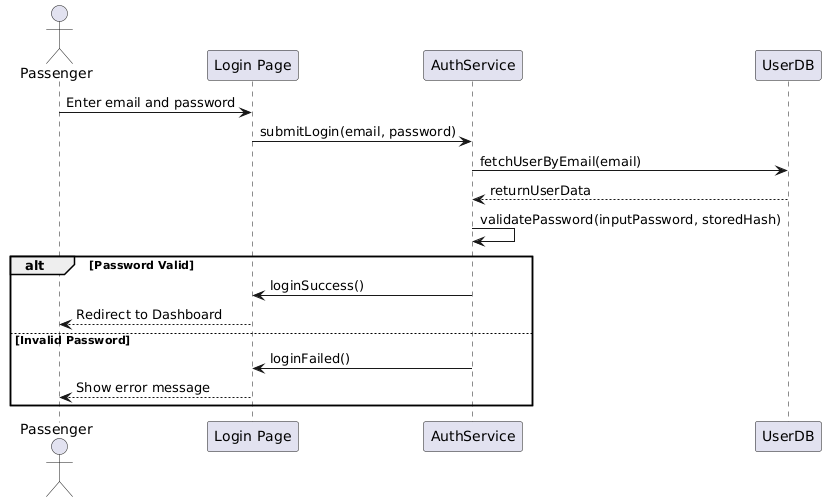
**5.5 Component Diagram**

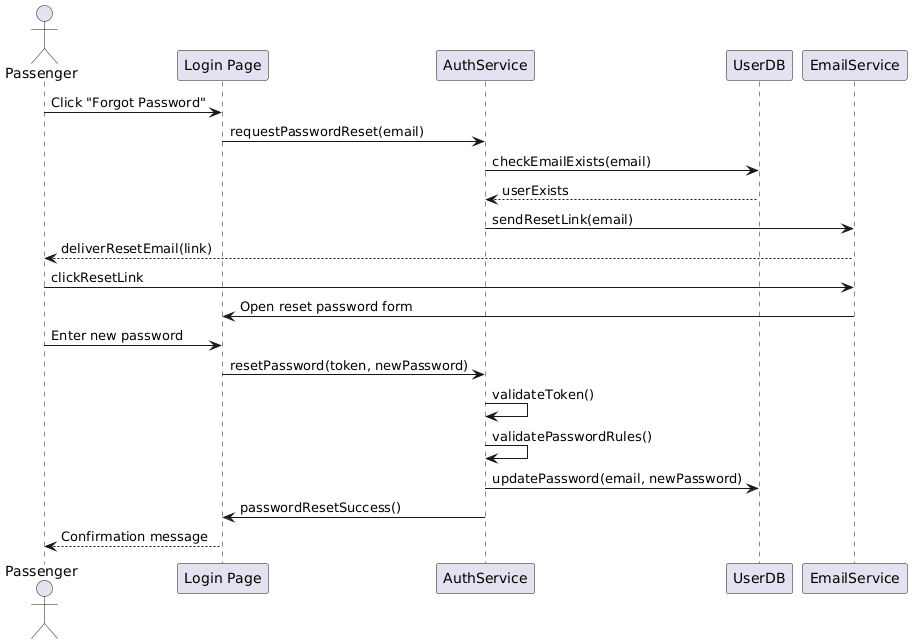


* 1. Sequence Diagrams

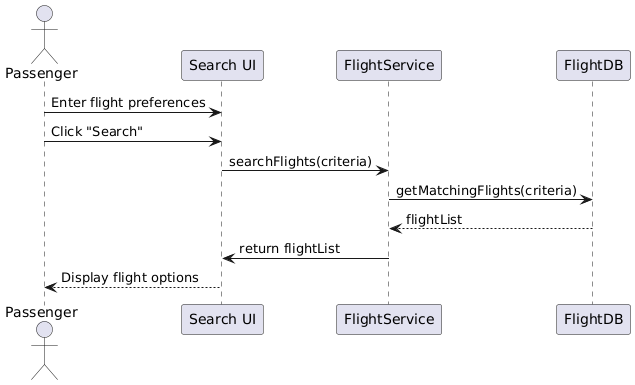
UC-101: User Registration – Sequence Diagram

UC-102: User Login – Sequence Diagram

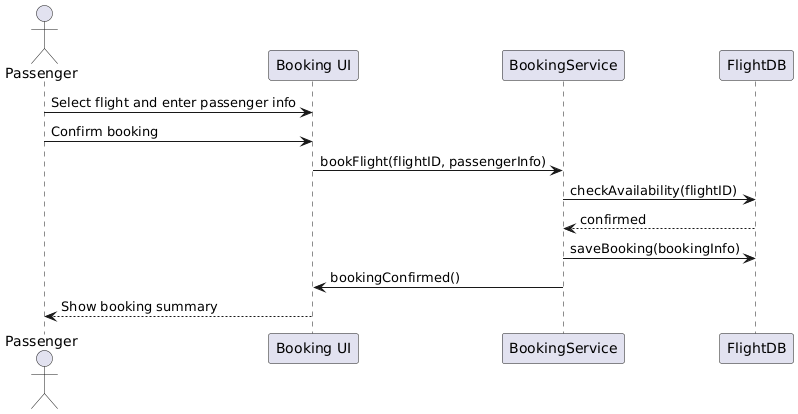


UC-103: Password Reset – Sequence Diagram

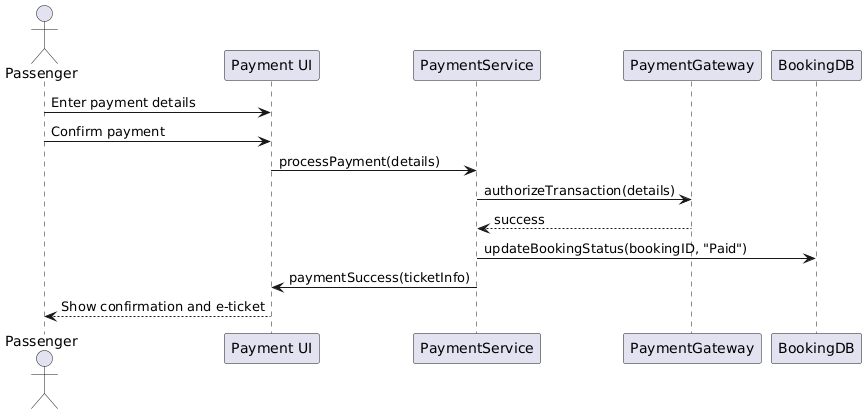
UC-201: Filter Flights – Sequence Diagram



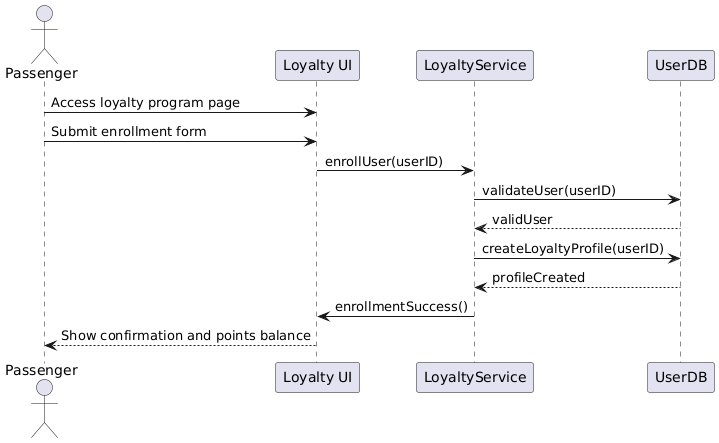
UC-301: Book Flight – Sequence Diagram



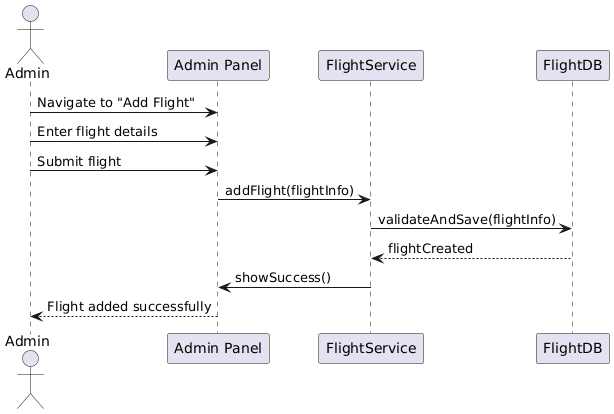
UC-302: Payment Processing – Sequence Diagram



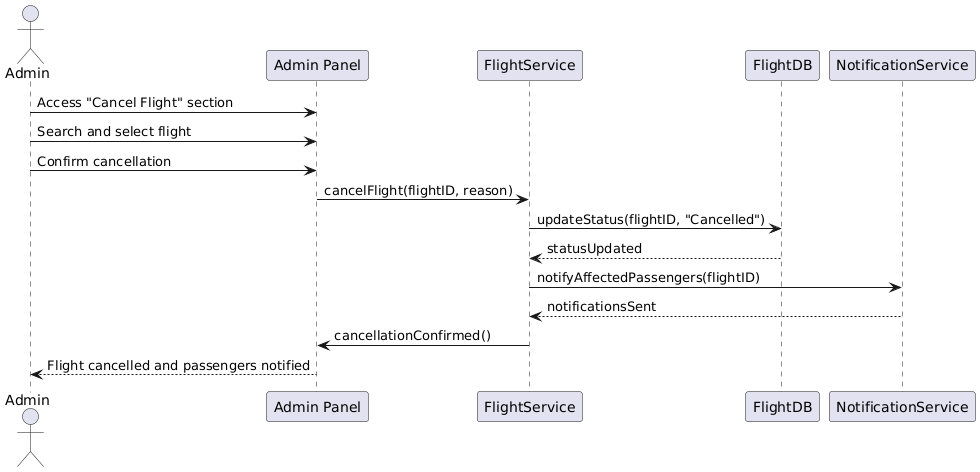
UC-401: Loyalty Program Enrollment – Sequence Diagram

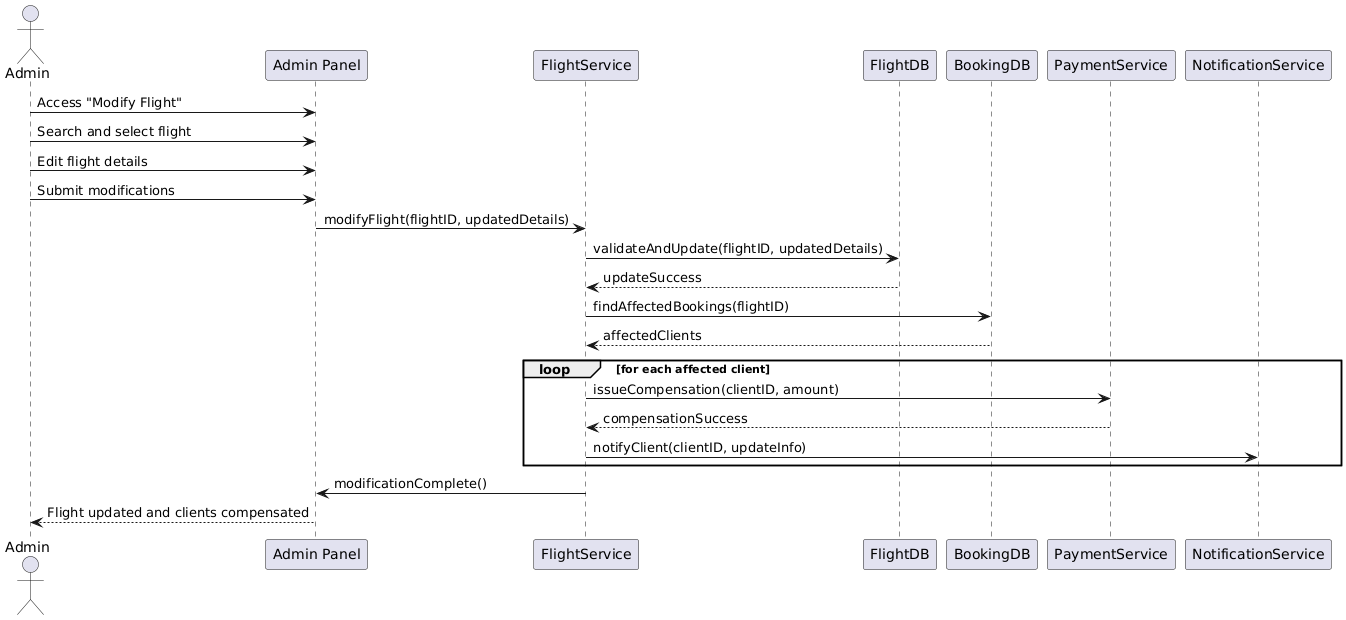


UC-501: Add Flight – Sequence Diagram

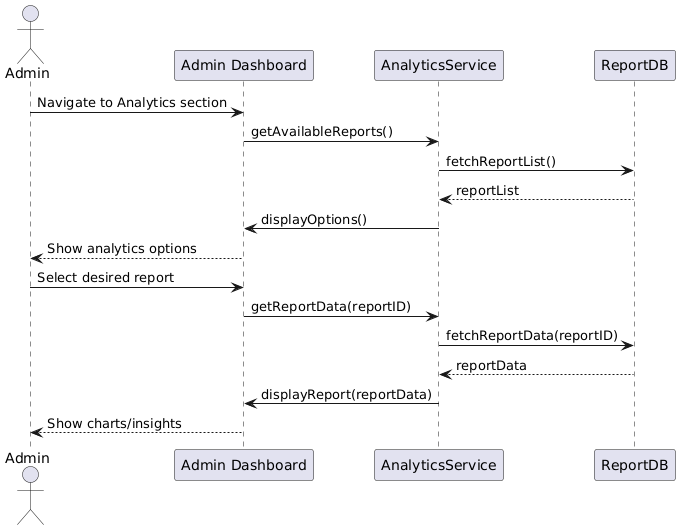


UC-502: Cancel Flight – Sequence Diagram

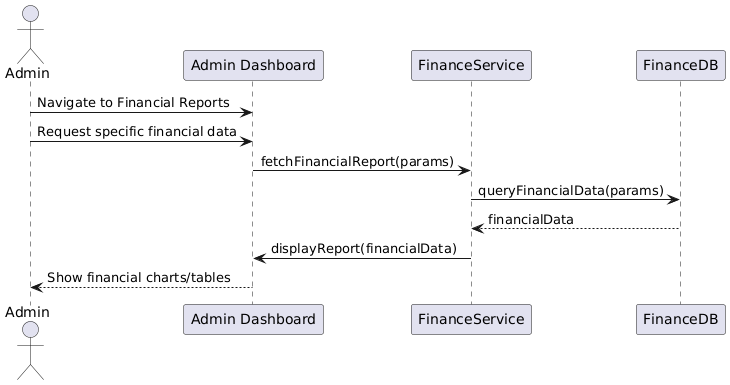


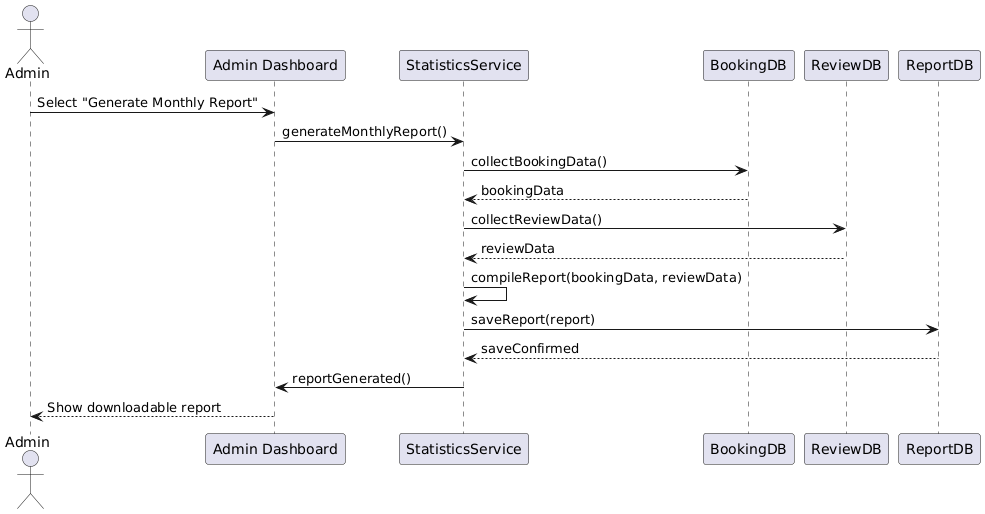
UC-503: Modify Flight & Issue Compensation – Sequence Diagram

UC-504: Analytics Access – Sequence Diagram



UC-505: Generate Monthly Statistical Reports – Sequence Diagram





UC-506: Access Financial Reports – Sequence Diagram

UC-601: Software Functionality – Tech Support Troubleshooting – Sequence Diagram

