Sistema de cambio de itinerarios en reservas de vuelos

Funcionalidades básicas de un sistema de gestión de reservas de aerolíneas, ordenadas por prioridad:

1. **Gestión de inventario de vuelos**
    a. Control de asientos disponibles en tiempo real
    b. Gestión de rutas y frecuencias
    c. Control de capacidad por clase (económica, business, primera)
2. **Motor de reservas**
    a. Búsqueda de vuelos disponibles
    b. Proceso de reserva de asientos
    c. Cálculo dinámico de tarifas
    d. Gestión de conexiones entre vuelos
3. **Gestión de pasajeros**
    a. Registro de datos personales
    b. Documentación de viaje
    c. Historial de reservas
    d. Programas de fidelización
4. **Sistema de pagos**
    a. Procesamiento de pagos
    b. Gestión de reembolsos
    c. Múltiples métodos de pago
    d. Facturación
5. **Check-in y gestión de embarque**
    a. Check-in online y en aeropuerto
    b. Asignación de asientos
    c. Gestión de equipaje
    d. Emisión de tarjetas de embarque
6. **Gestión de cambios y cancelaciones**
    a. Modificación de reservas
    b. Políticas de cancelación
    c. Cálculo de penalizaciones
    d. Reubicación en otros vuelos
7. **Reporting y análisis**
    a. Ocupación de vuelos

    b. Ingresos por ruta
    c. Análisis de tendencias
    d. Reportes operativos
8. **Integración con otros sistemas**
    a. GDS (Global Distribution Systems)
    b. Sistemas de control de operaciones
    c. APIs para terceros
    d. Sistemas de meteorología
9. **Gestión de servicios adicionales**
    a. Selección de comidas
    b. Servicios especiales
    c. Upgrade de clase
    d. Venta de servicios auxiliares
10. **Comunicación con pasajeros**
    a. Notificaciones de vuelo
    b. Alertas de cambios
    c. Confirmaciones de reserva
    d. Comunicación de incidencias

Principales beneficios para el cliente al usar un sistema de gestión de reservas de aerolíneas:

1. **Conveniencia y Accesibilidad**
    a. Reservas 24/7 desde cualquier dispositivo
    b. Eliminación de necesidad de visitar agencias físicas
    c. Acceso inmediato a toda la información de vuelos
    d. Gestión autónoma de sus reservas
2. **Ahorro de Tiempo**
    a. Proceso de reserva rápido y simplificado
    b. Check-in online anticipado
    c. Comparación instantánea de opciones de vuelo
    d. Actualización automática de información
3. **Ahorro de Dinero**
    a. Acceso a ofertas y promociones especiales
    b. Comparación fácil de tarifas
    c. Visibilidad de fechas con mejores precios
    d. Acumulación de puntos/millas de fidelización
4. **Mayor Control y Flexibilidad**

a. Selección de asientos en tiempo real
b. Modificación de reservas online
c. Gestión de preferencias personales
d. Cancelaciones autoservicio

5. **Mejor Experiencia de Viaje**
    a. Notificaciones proactivas sobre el vuelo
    b. Información actualizada sobre cambios
    c. Gestión digital de documentos de viaje
    d. Personalización de servicios adicionales
6. **Transparencia**
    a. Información clara sobre precios y condiciones
    b. Desglose detallado de costos
    c. Visibilidad del estado de la reserva
    d. Políticas de cambio y cancelación claras
7. **Seguridad y Confianza**
    a. Confirmación inmediata de reservas
    b. Almacenamiento seguro de datos personales
    c. Historial de transacciones accesible
    d. Pagos seguros online
8. **Servicios Personalizados**
    a. Recomendaciones basadas en preferencias
    b. Ofertas personalizadas
    c. Recordatorios automáticos
    d. Atención a necesidades especiales
9. **Integración con Otros Servicios**
    a. Reserva de servicios adicionales (hotel, coche)
    b. Conexión con programas de fidelización
    c. Integración con calendarios personales
    d. Compartir itinerarios fácilmente
10. **Soporte Post-Venta**
    a. Acceso a atención al cliente 24/7
    b. Gestión de reclamaciones online
    c. Seguimiento de equipaje
    d. Feedback y valoraciones

Estos beneficios contribuyen a:

- Mejorar la satisfacción del cliente

- Aumentar la lealtad a la aerolínea
- Reducir la fricción en el proceso de compra
- Proporcionar una experiencia de viaje más fluida y agradable

Customer journey típico de un cliente que utiliza un sistema de gestión de reservas de aerolíneas, desde la planificación hasta el post-viaje:

1. Fase de Planificación

- **Búsqueda inicial**
  - Ingreso al sitio web/app de la aerolínea
  - Introducción de criterios de búsqueda (origen, destino, fechas, pasajeros)
  - Exploración de diferentes opciones de vuelos
  - Comparación de precios y horarios
- **Selección de vuelo**
  - Revisión de detalles del vuelo (duración, escalas, tipo de avión)
  - Verificación de tarifas y condiciones
  - Selección de clase de viaje
  - Revisión de servicios incluidos

2. Fase de Reserva

- **Proceso de reserva**
  - Selección de asientos
  - Ingreso de datos de pasajeros
  - Añadir servicios adicionales (equipaje extra, comidas especiales)
  - Selección de método de pago
- **Confirmación**
  - Revisión final de la reserva
  - Proceso de pago
  - Recepción de confirmación por email
  - Almacenamiento de reserva en cuenta personal

3. Pre-vuelo

- **Preparación**
  - Recepción de recordatorios del vuelo
  - Check-in online (24-48h antes)

    - Selección/confirmación de asiento
    - Descarga de tarjeta de embarque
- **Gestión de cambios (si necesario)**
    - Modificación de horarios/fechas
    - Actualización de servicios especiales
    - Gestión de cancelaciones
    - Solicitud de reembolsos

4. Día del Vuelo

- **En el aeropuerto**
    - Check-in en mostrador (si no se hizo online)
    - Facturación de equipaje
    - Paso por seguridad usando boarding pass
    - Monitoreo de estado del vuelo
- **Pre-embarque**
    - Localización de puerta de embarque
    - Recepción de actualizaciones de vuelo
    - Gestión de cambios de última hora
    - Embarque según grupo asignado

5. Durante el Vuelo

- **Servicios a bordo**
    - Acceso a servicios reservados
    - Compras a bordo
    - Uso de entretenimiento
    - Registro de incidencias

6. Post-Vuelo

- **Llegada**
    - Recogida de equipaje
    - Notificación de llegada de equipaje
    - Reporte de incidencias (si las hay)
- **Seguimiento**
    - Acumulación de puntos/millas
    - Encuesta de satisfacción

    - Actualización del historial de viajes
    - Recepción de ofertas personalizadas

Puntos de Contacto Digital

Durante todo el journey, el cliente interactúa con:

- Sitio web de la aerolínea
- Aplicación móvil
- Emails automatizados
- SMS/Notificaciones push
- Quioscos de autoservicio
- Pantallas de información
- Chatbots/Asistentes virtuales

Momentos Críticos

Los momentos más importantes donde el sistema debe ser especialmente eficiente:

1. Búsqueda y reserva inicial
2. Proceso de pago
3. Check-in online
4. Notificaciones de cambios
5. Gestión de incidencias

Este journey puede variar según:

- Tipo de viaje (doméstico/internacional)
- Política de la aerolínea
- Nivel de digitalización
- Servicios contratados
- Programa de fidelización del cliente

Casos de uso más importantes para un MVP de gestión de cambios de itinerario, ordenados por prioridad y agrupados por funcionalidad:

1. Consulta de Reserva

1. **Buscar Reserva**

   a. Input: Código de reserva + Apellido/Email
   b. Output: Detalles de la reserva actual
   c. Validación de que la reserva existe y está activa
2. **Verificar Elegibilidad de Cambio**
   a. Validar si la reserva permite cambios
   b. Verificar estado del vuelo (no iniciado/completado)
   c. Comprobar restricciones de la tarifa

2. Búsqueda de Alternativas

3. **Buscar Vuelos Alternativos**
   a. Por cambio de fecha (mismo origen-destino)
   b. Por cambio de ruta (nuevo destino)
   c. Mostrar disponibilidad y tarifas
4. **Calcular Diferencias de Tarifa**
   a. Cálculo de diferencial de precio
   b. Cálculo de penalizaciones aplicables
   c. Mostrar costo total del cambio

3. Proceso de Cambio

5. **Seleccionar Nueva Opción**
   a. Selección del nuevo vuelo
   b. Confirmación de nuevos horarios
   c. Vista previa del cambio
6. **Procesar Pago de Diferencias**
   a. Gestión del pago adicional (si aplica)
   b. Proceso de reembolso (si aplica)
   c. Confirmación de transacción
7. **Confirmar Cambio**
   a. Actualización de la reserva
   b. Generación de nuevo itinerario
   c. Cancelación del vuelo anterior

4. Notificaciones

8. **Enviar Confirmaciones**
   a. Email de confirmación del cambio

b. Nuevo itinerario en PDF
c. Comprobante de cargo/reembolso

5. Gestión de Errores

9. **Manejar Fallos**
    a. Reversión de cambios fallidos
    b. Notificación de errores
    c. Log de intentos fallidos

Consideraciones Técnicas Importantes:

- Manejo de concurrencia en reservas
- Consistencia en el inventario de asientos
- Registro de auditoría de cambios
- Timeout en sesiones de cambio
- Validación de reglas de negocio
- Integración con sistema de pagos
- Gestión de estados de la transacción

Datos Mínimos Necesarios:

1. **Reserva**
    a. Código de reserva
    b. Estado
    c. Pasajeros
    d. Vuelos actuales
    e. Tarifa aplicada
2. **Vuelo**
    a. Número de vuelo
    b. Origen/Destino
    c. Fecha/Hora
    d. Disponibilidad
    e. Precios
3. **Cambio**
    a. ID de transacción
    b. Tipo de cambio
    c. Diferencial de precio

d. Estado del cambio
e. Timestamp

Este MVP proporcionaría la funcionalidad básica necesaria para gestionar cambios de itinerario, manteniendo la integridad de las reservas y asegurando una experiencia de usuario aceptable.

Diagrama de casos de uso en PlantUML que representa la funcionalidad del MVP, diferenciando entre usuarios visitantes y logueados:

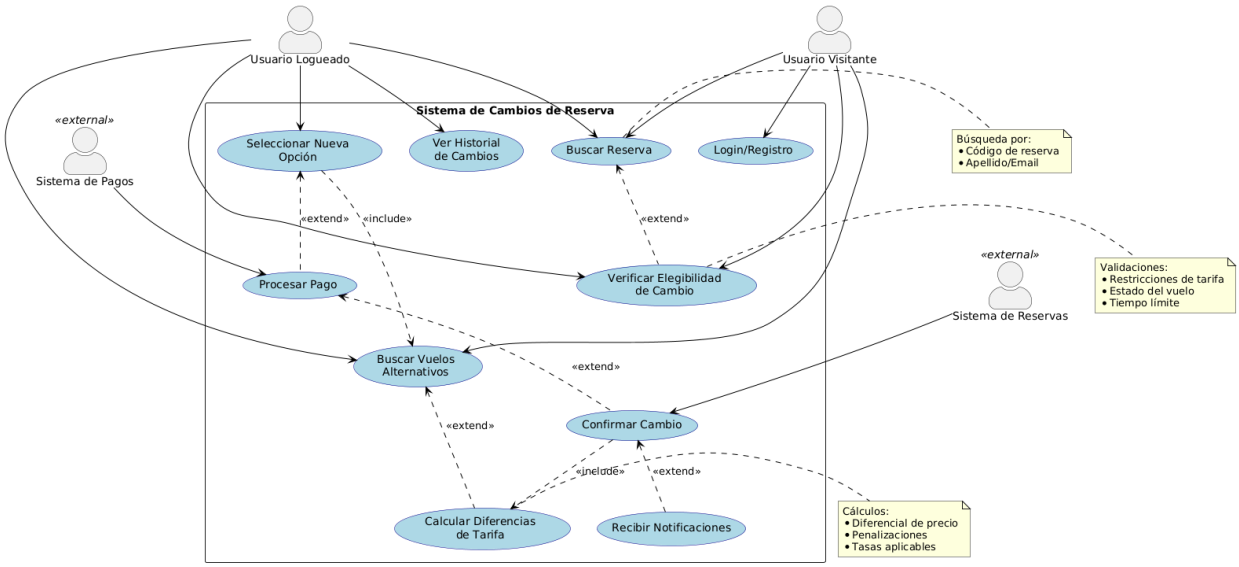
```
@startuml Airline Booking Change System

' Definición de estilos
skinparam actorStyle awesome
skinparam usecase {
  BackgroundColor LightBlue
  BorderColor DarkBlue
  ArrowColor Black
}

' Actores
actor "Usuario Visitante" as Guest
actor "Usuario Logueado" as User
actor "Sistema de Pagos" as PaymentSystem <<external>>
actor "Sistema de Reservas" as BookingSystem <<external>>

' Casos de uso principales
rectangle "Sistema de Cambios de Reserva" {
   usecase "Buscar Reserva" as UC1
   usecase "Verificar Elegibilidad\nde Cambio" as UC2
   usecase "Buscar Vuelos\nAlternativos" as UC3
   usecase "Calcular Diferencias\nde Tarifa" as UC4
   usecase "Seleccionar Nueva\nOpción" as UC5
   usecase "Procesar Pago" as UC6
   usecase "Confirmar Cambio" as UC7
   usecase "Recibir Notificaciones" as UC8
   usecase "Login/Registro" as UCAuth
   usecase "Ver Historial\nde Cambios" as UC9
```

```
    ' Relaciones de extensión
    UC1 <.. UC2 : <<extend>>
    UC3 <.. UC4 : <<extend>>
    UC5 <.. UC6 : <<extend>>
    UC6 <.. UC7 : <<extend>>
    UC7 <.. UC8 : <<extend>>

    ' Relaciones de inclusión
    UC5 ..> UC3 : <<include>>
    UC7 ..> UC4 : <<include>>
}

' Relaciones con actores
Guest --> UC1
Guest --> UC2
Guest --> UC3
Guest --> UCAuth

User --> UC1
User --> UC2
User --> UC3
User --> UC5
User --> UC9

PaymentSystem --> UC6
BookingSystem --> UC7

note right of UC1
  Búsqueda por:
  * Código de reserva
  * Apellido/Email
end note

note right of UC2
  Validaciones:
  * Restricciones de tarifa
  * Estado del vuelo
  * Tiempo límite
```

end note

note right of UC4
  Cálculos:
  * Diferencial de precio
  * Penalizaciones
  * Tasas aplicables
end note

@enduml



Este diagrama muestra:

1. **Actores**:
    a. Usuario Visitante: Puede realizar operaciones básicas sin autenticación
    b. Usuario Logueado: Tiene acceso a todas las funcionalidades
    c. Sistema de Pagos: Sistema externo para procesar pagos
    d. Sistema de Reservas: Sistema externo para gestionar reservas
2. **Casos de Uso**:
    a. Organizados en secuencia lógica de operación
    b. Con relaciones de extensión (<>) para funcionalidades opcionales
    c. Con relaciones de inclusión (<>) para funcionalidades requeridas
3. **Notas**:
    a. Detalles importantes sobre casos de uso clave

    b. Validaciones y cálculos relevantes
4. **Relaciones**:
    a. Muestra claramente qué puede hacer cada tipo de usuario
    b. Indica la interacción con sistemas externos
    c. Establece el flujo lógico de operaciones
5. **Agrupación**:
    a. Todo contenido en un sistema rectangular que representa el alcance del MVP
    b. Clara separación de responsabilidades

El diagrama sigue las buenas prácticas de UML:

- Nombres descriptivos y concisos
- Relaciones claramente definidas
- Uso apropiado de estereotipos
- Agrupación lógica de funcionalidades
- Inclusión de sistemas externos relevantes
- Notas explicativas en casos clave

Te ayudo con un modelo básico para un sistema de cambio de itinerarios aéreos. Aquí están las 3 entidades fundamentales:

1. **Reserva (Booking)**
    a. ID_Reserva (PK)
    b. ID_Cliente (FK)
    c. Estado (confirmada, pendiente, cancelada)
    d. Fecha_Creación
    e. Precio_Total
    f. Número_Pasajeros
    g. Código_Reserva (alfanumérico para el cliente)
2. **Vuelo (Flight)**
    a. ID_Vuelo (PK)
    b. Número_Vuelo
    c. ID_Aerolínea
    d. Origen
    e. Destino
    f. Fecha_Salida
    g. Hora_Salida
    h. Fecha_Llegada

i. Hora_Llegada
j. Capacidad_Total
k. Asientos_Disponibles
l. Estado (programado, retrasado, cancelado)

3. **Segmento_Reserva (BookingSegment)**
    a. ID_Segmento (PK)
    b. ID_Reserva (FK)
    c. ID_Vuelo (FK)
    d. Estado_Segmento
    e. Número_Asiento
    f. Clase_Servicio (económica, business, primera)
    g. Precio_Segmento

Relaciones:

- Una **Reserva** puede tener múltiples **Segmentos_Reserva** (relación 1:N) para manejar vuelos con escalas
- Cada **Segmento_Reserva** está asociado a un único **Vuelo** (relación N:1)
- La entidad **Segmento_Reserva** actúa como una tabla de unión que permite rastrear qué vuelos específicos forman parte de una reserva

Esta estructura permite:

- Gestionar cambios de itinerario modificando los segmentos
- Mantener un historial de cambios
- Manejar reservas con múltiples vuelos
- Controlar la disponibilidad de asientos
- Calcular diferencias de precio en cambios de itinerario

Las 3 entidades fundamentales para un sistema de cambio de itinerarios en reservas aerolíneas, sus campos esenciales y sus relaciones:

1. **Reserva (Booking)**
    a. booking_id (Identificador único)
    b. passenger_id (Relación con el pasajero)
    c. booking_status (Confirmada, Pendiente, Cancelada)
    d. booking_date (Fecha de la reserva)
    e. total_amount (Monto total)
    f. currency (Moneda)

g. payment_status (Estado del pago)

2. **Itinerario (Itinerary)**
    a. itinerary_id (Identificador único)
    b. booking_id (Relación con la reserva)
    c. original_flight_id (Vuelo original)
    d. new_flight_id (Vuelo nuevo en caso de cambio)
    e. change_fee (Cargo por cambio)
    f. change_status (Pendiente, Aprobado, Rechazado)
    g. change_date (Fecha del cambio)
    h. reason (Motivo del cambio)
3. **Vuelo (Flight)**
    a. flight_id (Identificador único)
    b. flight_number (Número de vuelo)
    c. origin_airport (Aeropuerto de salida)
    d. destination_airport (Aeropuerto de llegada)
    e. departure_datetime (Fecha y hora de salida)
    f. arrival_datetime (Fecha y hora de llegada)
    g. available_seats (Asientos disponibles)
    h. fare_class (Clase de tarifa)
    i. aircraft_id (Identificador de la aeronave)

Relaciones entre las entidades:

1. **Reserva - Itinerario (1:N)**
    a. Una reserva puede tener múltiples itinerarios (en caso de vuelos con escalas o cambios históricos)
    b. Cada itinerario pertenece a una única reserva
2. **Itinerario - Vuelo (N:2)**
    a. Cada itinerario está relacionado con hasta dos vuelos:
        i. El vuelo original
        ii. El vuelo nuevo (en caso de cambio)
    b. Un vuelo puede estar asociado a múltiples itinerarios
3. **Reserva - Vuelo (M:N)**
    a. Una reserva puede incluir múltiples vuelos
    b. Un vuelo puede estar asociado a múltiples reservas

Consideraciones adicionales:

- Es importante mantener un historial de cambios en los itinerarios para auditoría

- Se deben considerar las políticas de cambio y tarifas asociadas
- El sistema debe manejar la disponibilidad de asientos en tiempo real
- Se debe considerar la gestión de listas de espera para vuelos llenos
- Es crucial mantener la integridad referencial entre las entidades

Este modelo de datos proporciona la base necesaria para gestionar de manera eficiente los cambios de itinerario, manteniendo la trazabilidad y permitiendo una gestión efectiva de las modificaciones en las reservas.

Entiendo que ya tienes definido el modelo de datos base y ahora necesitas una visión más completa de la arquitectura del sistema. Te ayudaré a diseñar una arquitectura robusta para el sistema de cambio de itinerarios.

1. Arquitectura General (Patrón de Microservicios)

Propongo los siguientes servicios principales:

1. **Booking Service (Servicio de Reservas)**
    a. Gestión del ciclo de vida de las reservas
    b. Validación de disponibilidad
    c. Cálculo de tarifas y cargos
    d. API endpoints:
        i. POST /bookings (crear reserva)
        ii. GET /bookings/{id} (consultar reserva)
        iii. PUT /bookings/{id}/status (actualizar estado)
2. **Itinerary Change Service (Servicio de Cambios)**
    a. Gestión de solicitudes de cambio
    b. Validación de reglas de negocio para cambios
    c. Cálculo de diferencias tarifarias
    d. API endpoints:
        i. POST /changes/request (solicitar cambio)
        ii. GET /changes/{id} (consultar estado del cambio)
        iii. PUT /changes/{id}/confirm (confirmar cambio)
3. **Flight Inventory Service (Servicio de Inventario)**
    a. Gestión de disponibilidad de vuelos
    b. Control de asientos
    c. Bloqueo temporal de inventario
    d. API endpoints:
        i. GET /flights/availability

        ii. POST /flights/hold
        iii. PUT /flights/release
4. **Payment Service (Servicio de Pagos)**
    a. Procesamiento de pagos
    b. Reembolsos
    c. Cargos por cambio
    d. API endpoints:
        i. POST /payments
        ii. POST /refunds
        iii. GET /payments/{id}

2. Componentes de Infraestructura

- **API Gateway**
    - Enrutamiento
    - Rate limiting
    - Autenticación/Autorización
    - Logging
- **Message Broker (Kafka/RabbitMQ)**
    - Comunicación asíncrona entre servicios
    - Eventos de cambio de estado
    - Notificaciones
- **Cache Distribuida (Redis)**
    - Caché de disponibilidad
    - Bloqueos temporales
    - Sesiones de usuario
- **Base de Datos**
    - PostgreSQL para datos transaccionales
    - MongoDB para logs y auditoría

3. Flujo de Cambio de Itinerario

```
sequenceDiagram
  Cliente->>API Gateway: Solicita cambio
  API Gateway->>Itinerary Change: Valida solicitud
  Itinerary Change->>Flight Inventory: Verifica disponibilidad
  Flight Inventory-->>Itinerary Change: Confirma disponibilidad
  Itinerary Change->>Payment: Calcula cargos
  Payment-->>Itinerary Change: Retorna cargos
```

Itinerary Change-->>Cliente: Presenta opciones
Cliente->>API Gateway: Confirma cambio
API Gateway->>Payment: Procesa pago
Payment-->>Itinerary Change: Confirma pago
Itinerary Change->>Booking: Actualiza reserva
Booking->>Message Broker: Emite evento
Message Broker->>Notification: Envía confirmación

4. Consideraciones de Diseño

1. **Resiliencia**
    a. Circuit breakers
    b. Retry policies
    c. Fallbacks
    d. Timeouts configurables
2. **Escalabilidad**
    a. Servicios stateless
    b. Caché distribuida
    c. Sharding de bases de datos
    d. Auto-scaling
3. **Consistencia**
    a. Saga pattern para transacciones distribuidas
    b. Compensating transactions
    c. Event sourcing
4. **Monitoreo**
    a. Distributed tracing
    b. Métricas de negocio
    c. Health checks
    d. Logs centralizados

5. Aspectos de Seguridad

- JWT para autenticación
- HTTPS/TLS para todas las comunicaciones
- Encriptación de datos sensibles
- Rate limiting por API key
- Auditoría de cambios

6. Consideraciones de Implementación

1. **Tecnologías Sugeridas**
    a. Backend: Spring Boot/Node.js
    b. Base de Datos: PostgreSQL
    c. Cache: Redis
    d. Message Broker: Kafka
    e. API Gateway: Kong/Netflix Zuul
2. **Prácticas DevOps**
    a. CI/CD pipelines
    b. Infrastructure as Code
    c. Containerization (Docker)
    d. Orchestration (Kubernetes)

¿Te gustaría que profundicemos en algún aspecto específico de la arquitectura? Podemos explorar en detalle:

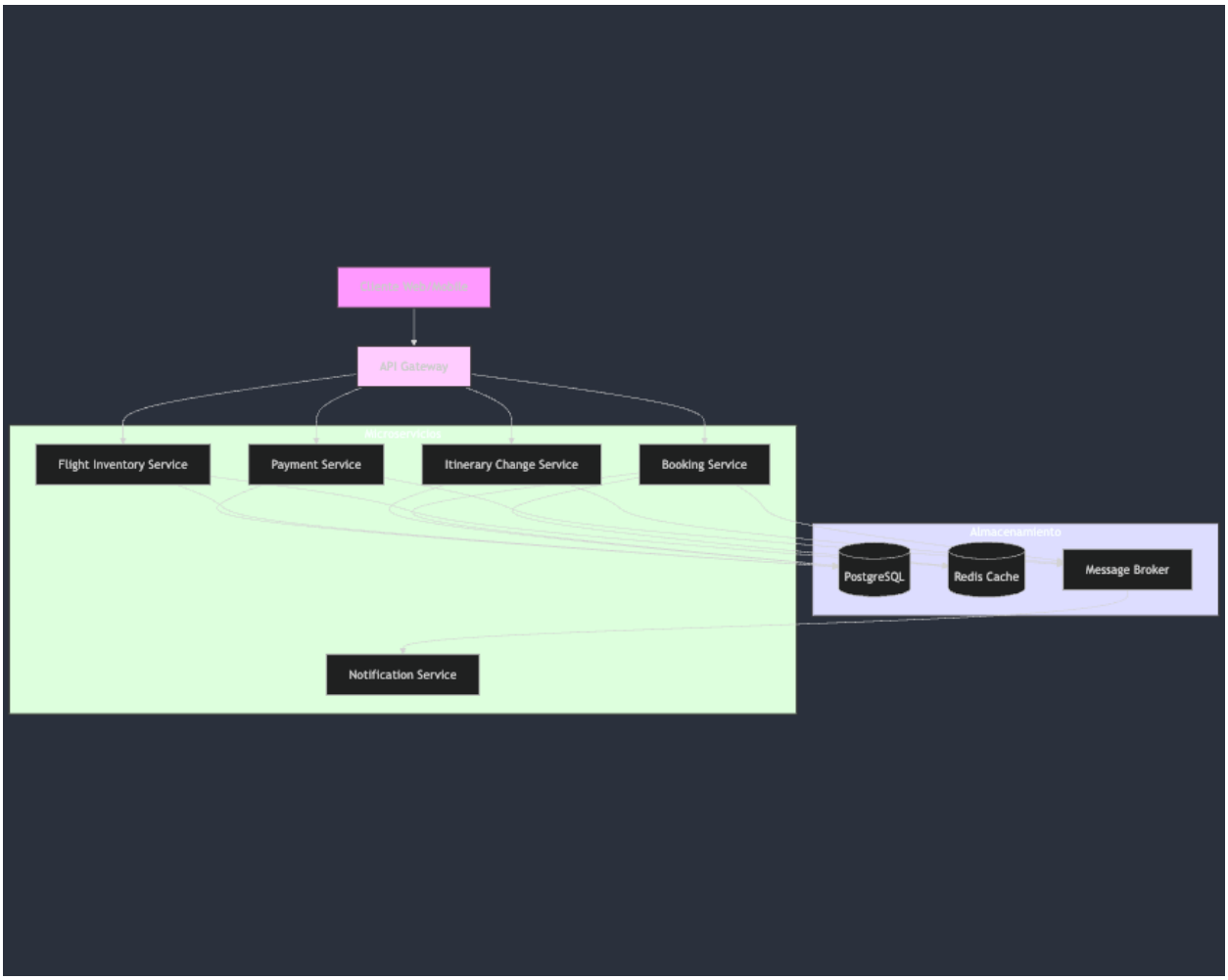
- Patrones de diseño específicos
- Flujos de datos
- Estrategias de caché
- Manejo de errores
- Estrategias de deployment
- Cualquier otro aspecto que consideres relevante

Te generaré varios diagramas Mermaid que representan diferentes vistas de la arquitectura del sistema.

1. Diagrama de Componentes y Servicios

```
graph TB
  Client[Cliente Web/Mobile]
  AG[API Gateway]

  subgraph Services[Microservicios]
    BS[Booking Service]
    ICS[Itinerary Change Service]
    FIS[Flight Inventory Service]
    PS[Payment Service]
    NS[Notification Service]
```

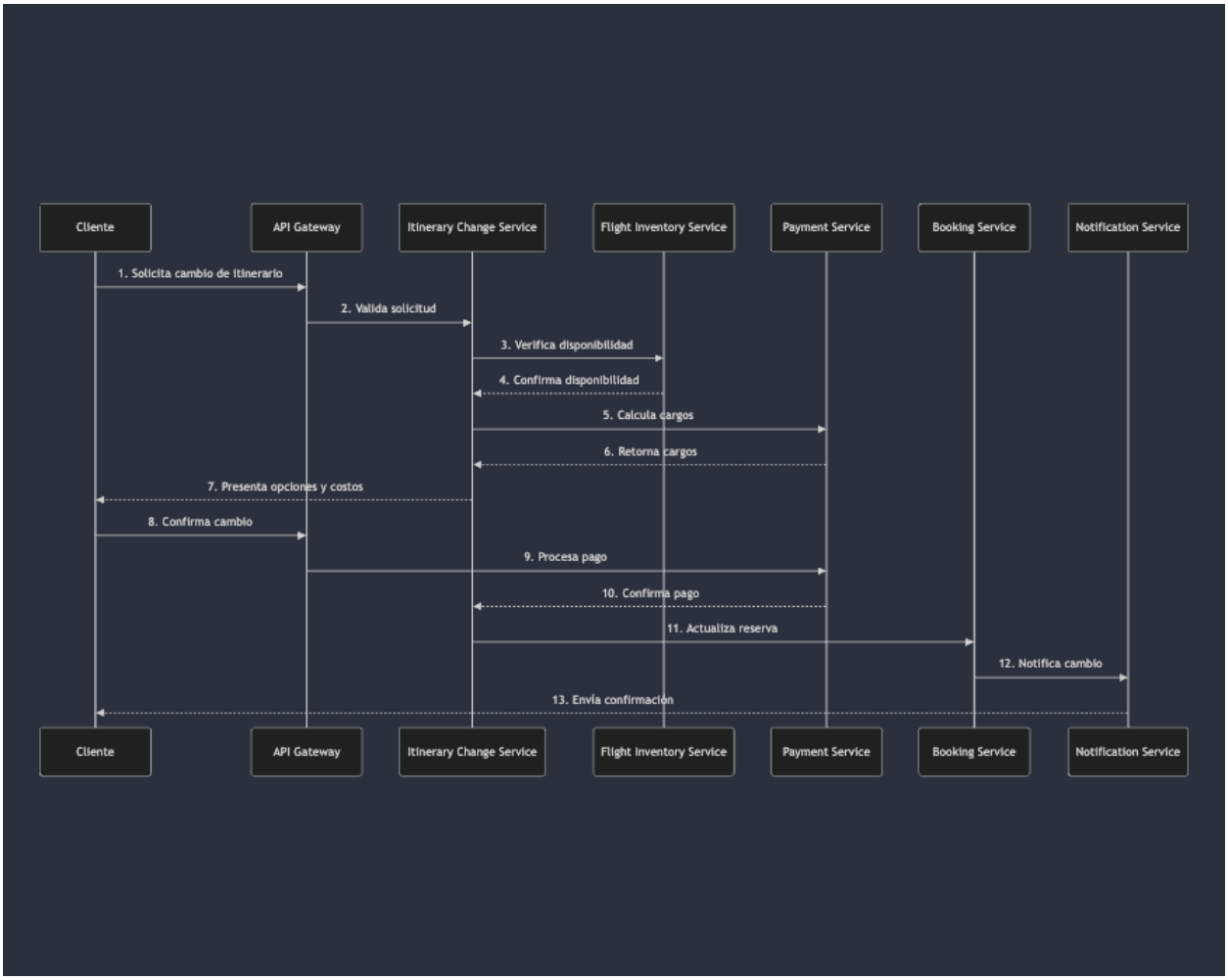
```
end

subgraph Data[Almacenamiento]
  DB[(PostgreSQL)]
  Cache[(Redis Cache)]
  MQ[Message Broker]
end

Client --> AG
AG --> BS
AG --> ICS
AG --> FIS
AG --> PS

BS --> DB
ICS --> DB
FIS --> DB
PS --> DB

BS --> Cache
FIS --> Cache

BS --> MQ
ICS --> MQ
PS --> MQ
MQ --> NS

style Client fill:#f9f,stroke:#333,stroke-width:2px
style AG fill:#fcf,stroke:#333,stroke-width:2px
style Services fill:#dfd,stroke:#333,stroke-width:2px
style Data fill:#ddf,stroke:#333,stroke-width:2px
```



2. Diagrama de Flujo de Cambio de Itinerario

```
sequenceDiagram
    participant C as Cliente
    participant AG as API Gateway
    participant ICS as Itinerary Change Service
    participant FIS as Flight Inventory Service
    participant PS as Payment Service
    participant BS as Booking Service
    participant NS as Notification Service

    C->>AG: 1. Solicita cambio de itinerario
    AG->>ICS: 2. Valida solicitud
    ICS->>FIS: 3. Verifica disponibilidad
```

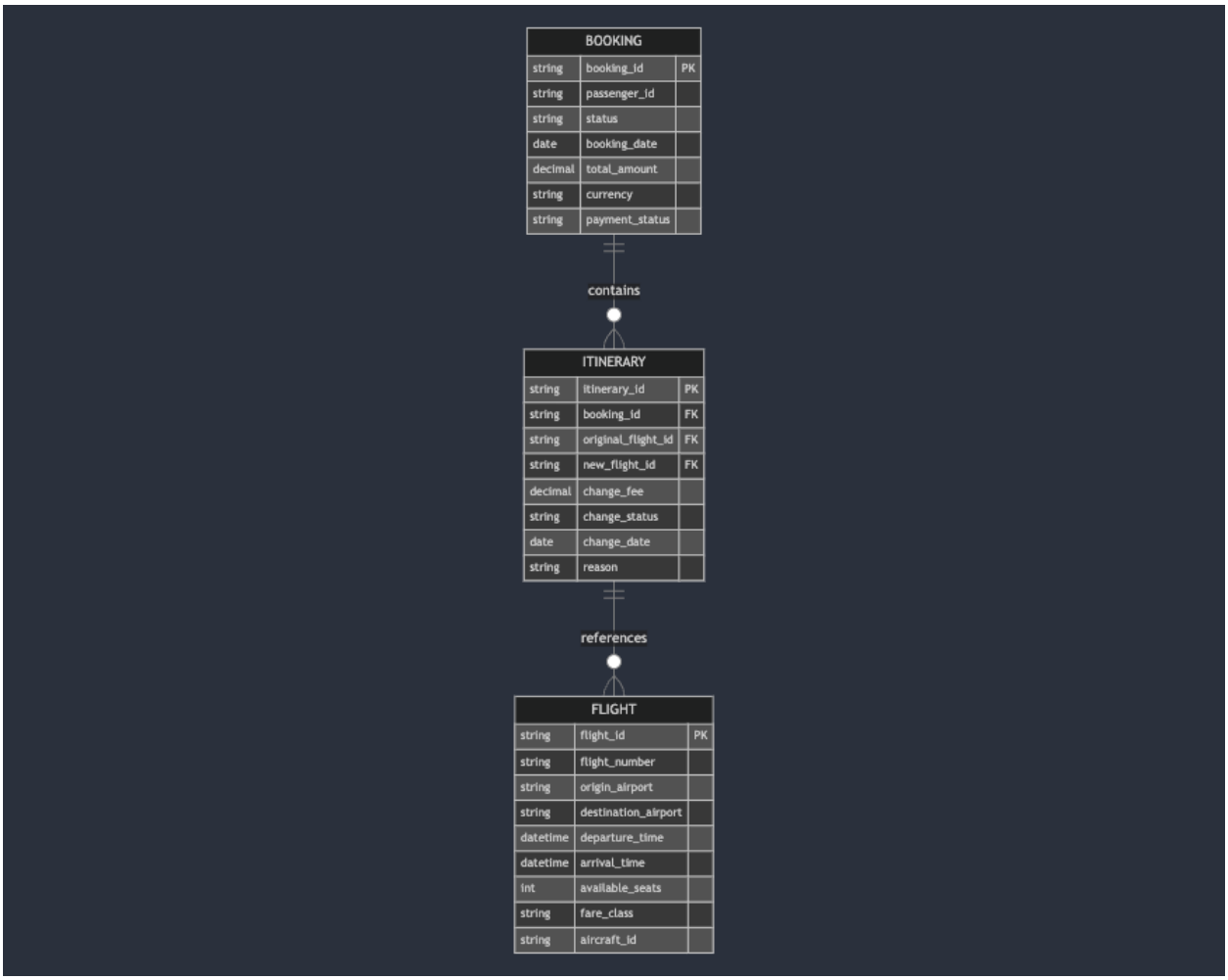
FIS-->>ICS: 4. Confirma disponibilidad
ICS->>PS: 5. Calcula cargos
PS-->>ICS: 6. Retorna cargos
ICS-->>C: 7. Presenta opciones y costos
C->>AG: 8. Confirma cambio
AG->>PS: 9. Procesa pago
PS-->>ICS: 10. Confirma pago
ICS->>BS: 11. Actualiza reserva
BS->>NS: 12. Notifica cambio
NS-->>C: 13. Envía confirmación



3. Diagrama de Modelo de Datos

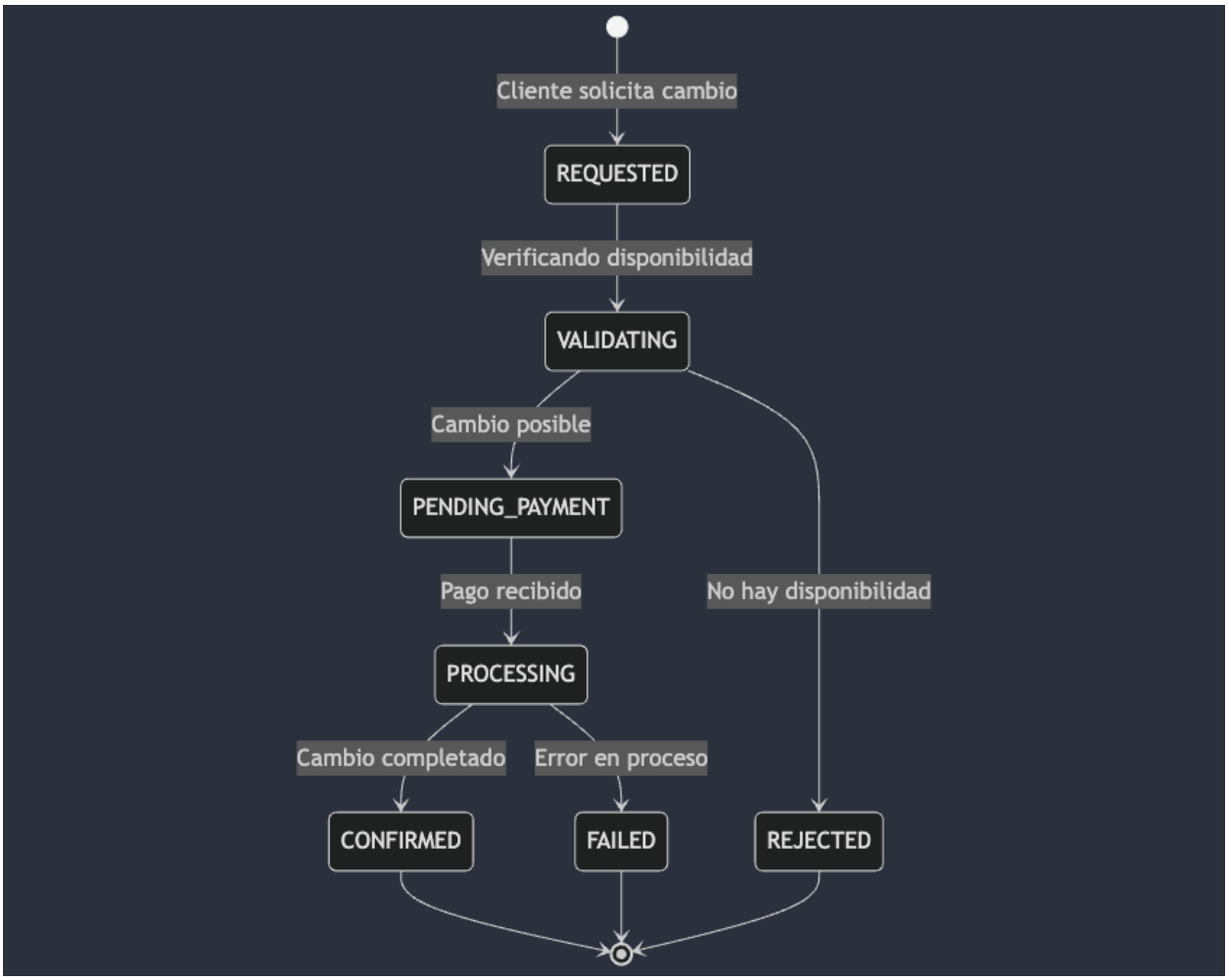
erDiagram
BOOKING ||--o{ ITINERARY : contains

```
BOOKING {
  string booking_id PK
  string passenger_id
  string status
  date booking_date
  decimal total_amount
  string currency
  string payment_status
}
ITINERARY ||--o{ FLIGHT : references
ITINERARY {
  string itinerary_id PK
  string booking_id FK
  string original_flight_id FK
  string new_flight_id FK
  decimal change_fee
  string change_status
  date change_date
  string reason
}
FLIGHT {
  string flight_id PK
  string flight_number
  string origin_airport
  string destination_airport
  datetime departure_time
  datetime arrival_time
  int available_seats
  string fare_class
  string aircraft_id
}
```



4. Diagrama de Estados de Cambio de Itinerario

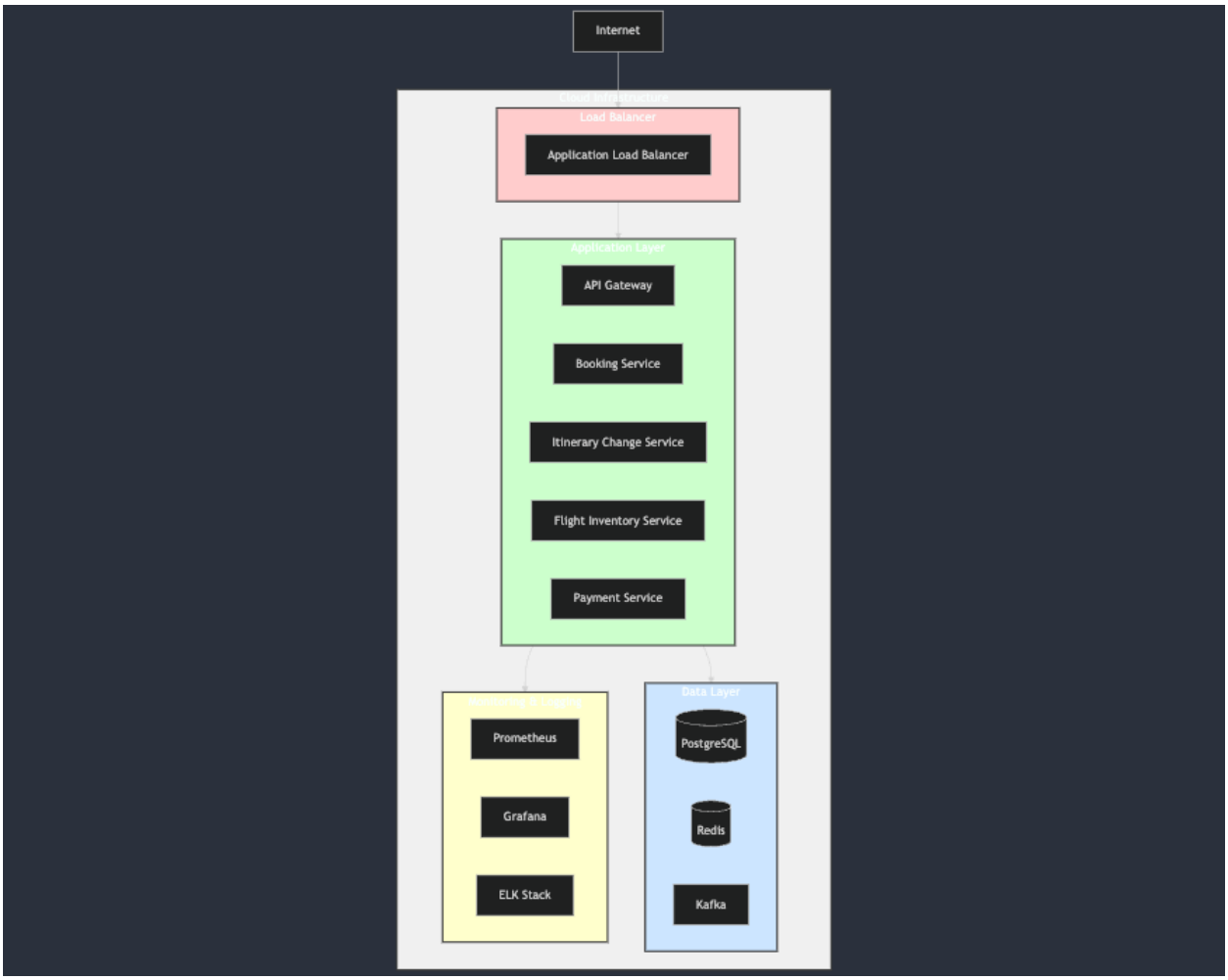
```
stateDiagram-v2
   [*] --> REQUESTED: Cliente solicita cambio
   REQUESTED --> VALIDATING: Verificando disponibilidad
   VALIDATING --> PENDING_PAYMENT: Cambio posible
   VALIDATING --> REJECTED: No hay disponibilidad
   PENDING_PAYMENT --> PROCESSING: Pago recibido
   PROCESSING --> CONFIRMED: Cambio completado
   PROCESSING --> FAILED: Error en proceso
   CONFIRMED --> [*]
   REJECTED --> [*]
   FAILED --> [*]
```



5. Diagrama de Despliegue

```
graph TB
  subgraph Cloud[Cloud Infrastructure]
    subgraph LB[Load Balancer]
      ALB[Application Load Balancer]
    end

    subgraph App[Application Layer]
      API[API Gateway]
      BS[Booking Service]
      ICS[Itinerary Change Service]
      FIS[Flight Inventory Service]
      PS[Payment Service]
    end
```

```
    subgraph Data[Data Layer]
        DB[(PostgreSQL)]
        Cache[(Redis)]
        MQ[Kafka]
    end

    subgraph Monitoring[Monitoring & Logging]
        PR[Prometheus]
        GF[Grafana]
        ELK[ELK Stack]
    end
end

Internet --> LB
LB --> App
App --> Data
App --> Monitoring

style Cloud fill:#f0f0f0,stroke:#333,stroke-width:2px
style LB fill:#ffcccc,stroke:#333,stroke-width:2px
style App fill:#ccffcc,stroke:#333,stroke-width:2px
style Data fill:#cce5ff,stroke:#333,stroke-width:2px
style Monitoring fill:#ffffcc,stroke:#333,stroke-width:2px
```



Estos diagramas proporcionan diferentes vistas de la arquitectura:

1. El diagrama de componentes muestra la estructura general del sistema
2. El diagrama de secuencia ilustra el flujo de un cambio de itinerario
3. El diagrama ERD muestra las relaciones entre las entidades principales
4. El diagrama de estados muestra el ciclo de vida de un cambio de itinerario
5. El diagrama de despliegue muestra la infraestructura cloud