



QUẢN LÝ CHUYẾN BAY






Thuyết trình bởi Nhóm 15.1



Gặp gỡ Nhóm

Hoàng Yến Nhi - 20225898
Nguyễn Khánh Duy - 20225830
Nguyễn Văn Phú - 20225658
Trần Lê Anh Minh - 20225652
Đỗ Tiến Đạt - 20225700

Mục lục

-  Mô tả dự án
-  Thiết kế cơ sở dữ liệu
-  Biểu đồ UML
-  Cấu trúc dự án
-  Demo



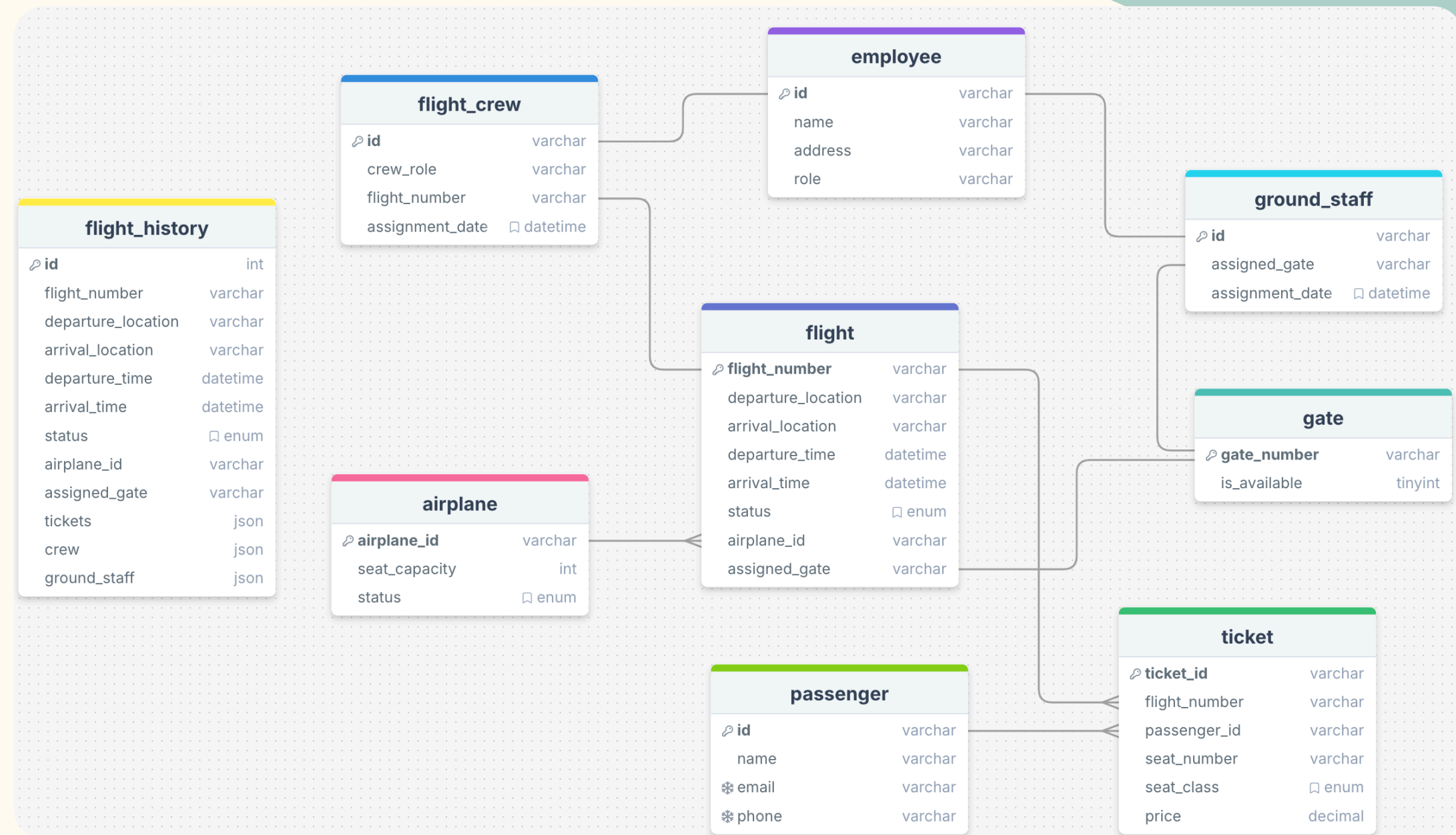
MÔ TẢ DỰ ÁN

Dự án Flight Management nhằm xây dựng một hệ thống quản lý chuyến bay với các chức năng chính liên quan đến quản lý chuyến bay, cổng (gate), hành khách, và vé. Dự án sử dụng Java và JavaFX để phát triển giao diện người dùng, JDBC để kết nối với cơ sở dữ liệu MySQL, và JSON để lưu trữ thông tin lịch sử chuyến bay.

Thiết kế CSDL

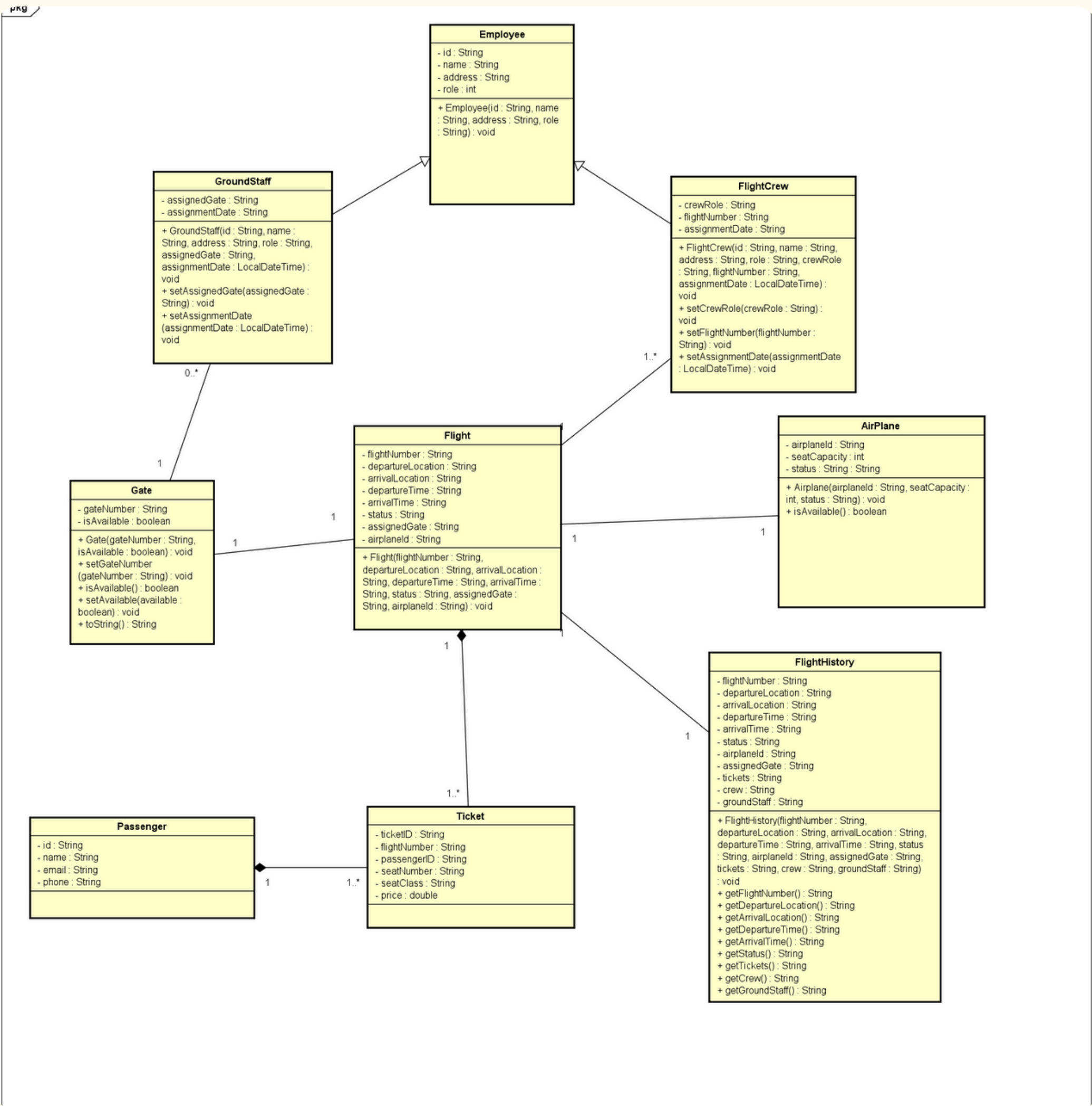
Quan hệ các bảng

- Liên kết giữa employee và flight_crew: 1-1
- Liên kết giữa employee và ground_staff: 1-1
- Liên kết giữa flight và airplane: 1-1
- Liên kết giữa flight và gate: 1-1
- Liên kết giữa flight và flight_crew: 1-n
- Liên kết giữa ground_staff và gate: 1-n
- Liên kết giữa flight và ticket: 1-n
- Liên kết giữa ticket và passenger: 1-1



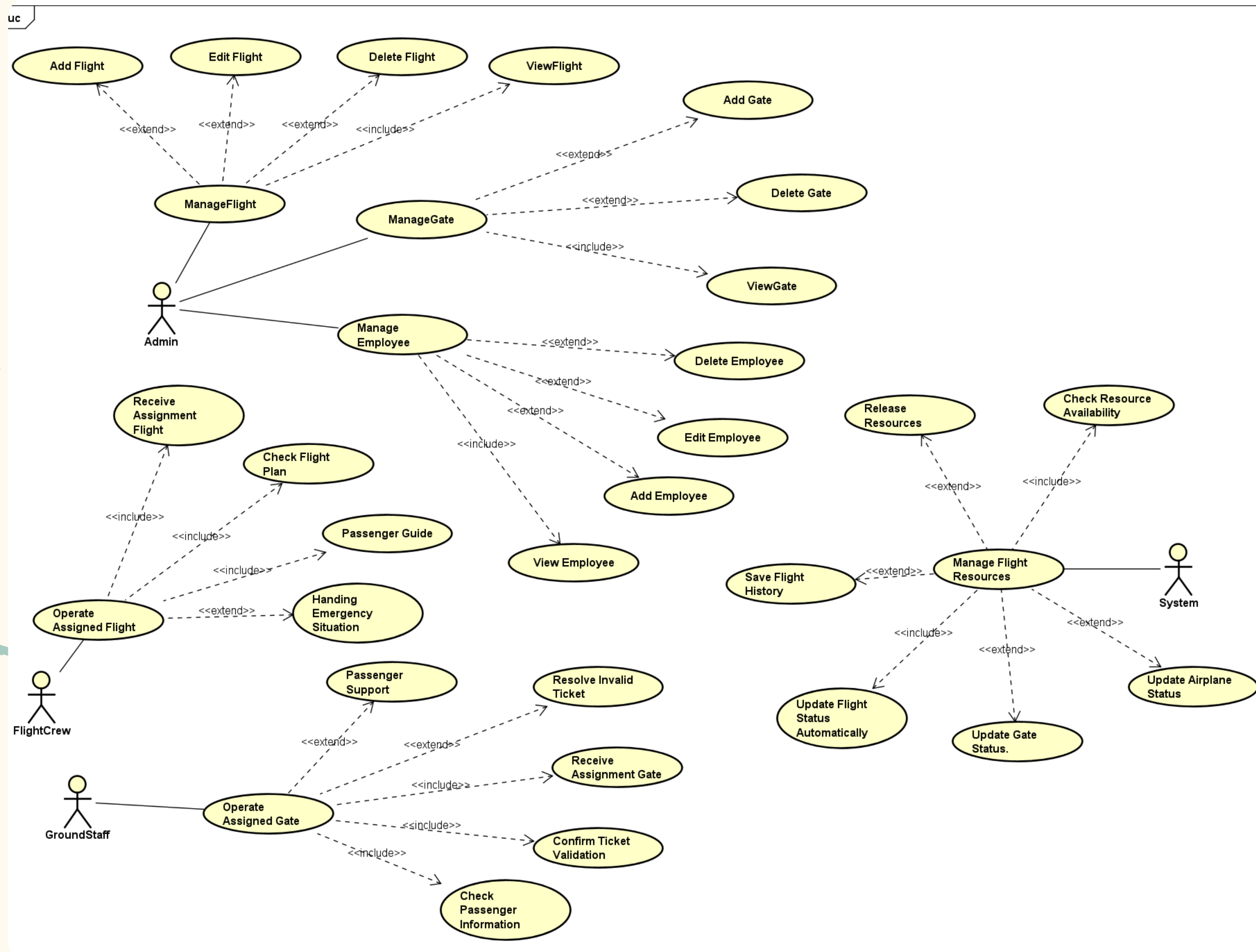
Biểu đồ UML

I. Biểu đồ Class



Biểu đồ UML

II. Biểu đồ UseCase



Model

Service

Controller

View

**Cấu trúc
dự án**

Model

- Tầng Model đại diện cho các thực thể (entities) trong hệ thống quản lý chuyến bay.
- Lưu trữ dữ liệu dưới dạng các thuộc tính và phương thức getter/setter.
- Mỗi lớp Model ánh xạ trực tiếp với một bảng trong cơ sở dữ liệu.
- Không chứa logic nghiệp vụ, chỉ quản lý dữ liệu thuần túy.

```
▼ models
  J Airplane.java
  J Employee.java
  J Flight.java
  J FlightCrew.java
  J FlightHistory.java
  J Gate.java
  J GroundStaff.java
  J Passenger.java
  J Ticket.java
  J TicketAdapter.java
```

Service

01 Tầng Service là trung tâm xử lý logic nghiệp vụ (business logic) trong hệ thống.

Thực hiện các logic nghiệp vụ phức tạp:

- 02**
- Kiểm tra dữ liệu từ người dùng (ví dụ: trạng thái cổng, số ghế trống).
 - Tính toán các giá trị cần thiết (ví dụ: tổng số lượng vé bán ra, doanh thu chuyến bay).

- 03**
- Truy vấn cơ sở dữ liệu:
- Thêm, sửa, xóa, hoặc tìm kiếm dữ liệu trực tiếp từ cơ sở dữ liệu MySQL.

- 04**
- Kết nối với tầng Controller:
- Nhận yêu cầu từ tầng Controller, xử lý dữ liệu, và trả kết quả về cho giao diện.

```
✓ services
  J AirplaneService.java
  J FlightCrewService.java
  J FlightHistoryService.java
  J FlightService.java
  J GateService.java
  J GroundStaffService.java
  J PassengerService.java
  J TicketService.java
```

Controller

- Tầng Controller điều khiển luồng dữ liệu giữa giao diện (View) và tầng Service.
- Quản lý các sự kiện từ giao diện (e.g., nhấn nút, nhập liệu).
- Không xử lý logic nghiệp vụ mà chỉ chuyển tiếp yêu cầu đến tầng Service.
- Tích hợp chặt chẽ với JavaFX để hiển thị dữ liệu và cập nhật giao diện.

▼ controllers

- J AddEmployeeController.java
- J AddFlightController.java
- J AssignEmployeeController.java
- J EditEmployeeController.java
- J EditFlightController.java
- J EmployeeController.java
- J FlightController.java
- J FlightDetailsController.java
- J FlightHistoryController.java
- J FlightHistoryDetailsController.java
- J ManageAirplaneController.java
- J ManageGateController.java

View

- 01** Tầng View là nơi giao tiếp giữa hệ thống và người dùng.

JavaFX Framework:

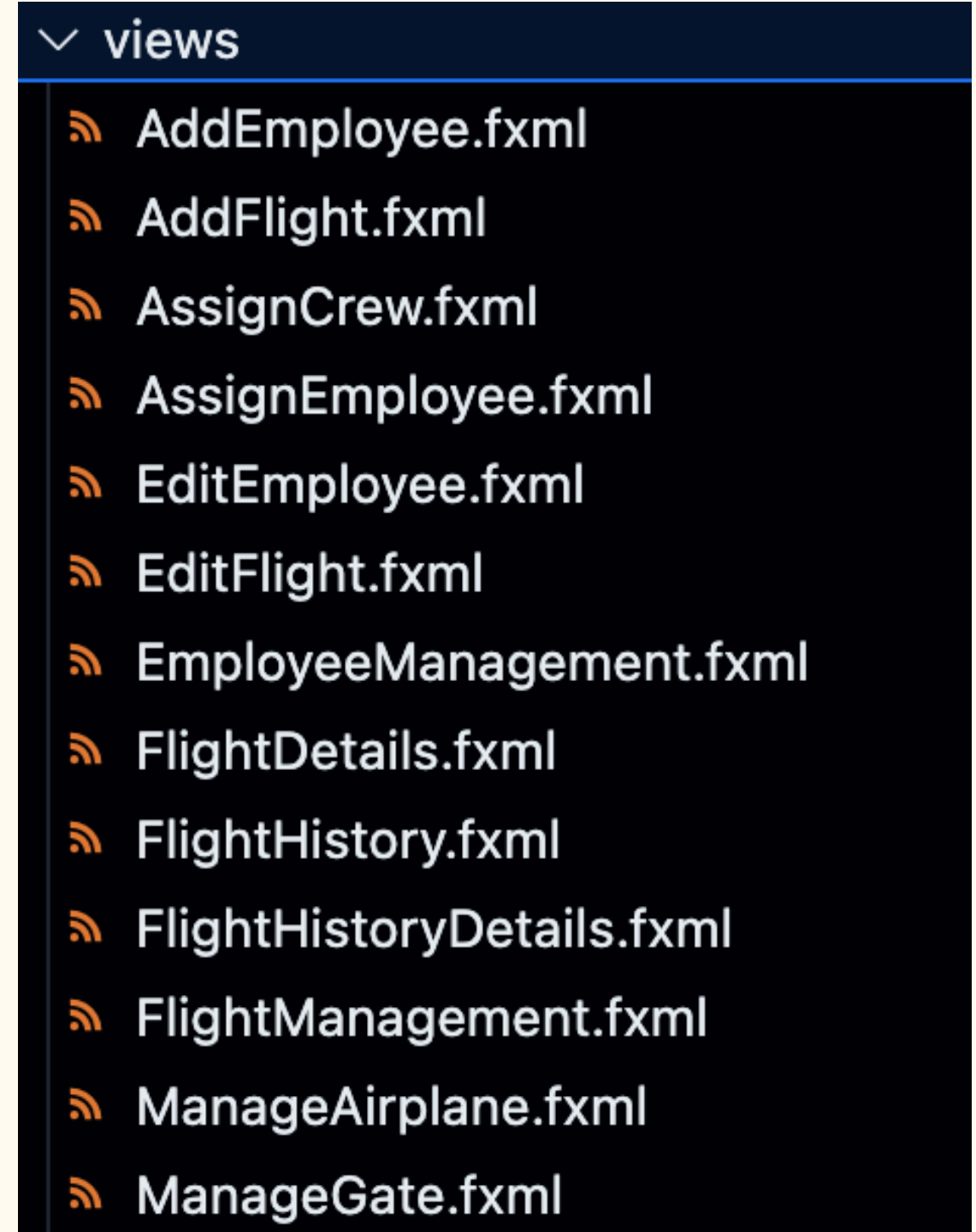
- 02**
- Cung cấp các thành phần giao diện như:
 - TableView: Hiển thị danh sách
 - Form: Các trường nhập liệu để thêm hoặc chỉnh sửa dữ liệu.
 - Button: Tạo các nút thao tác (e.g., thêm, sửa, xóa).
 - Hỗ trợ FXML để định nghĩa giao diện dưới dạng XML.

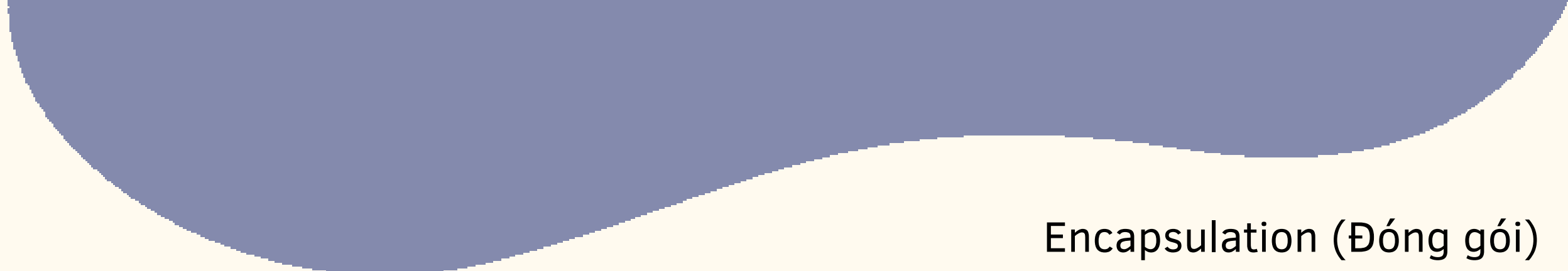
Binding Dữ Liệu:

- 03**
- Sử dụng các thuộc tính như StringProperty, IntegerProperty để liên kết dữ liệu giữa tầng Model và View.

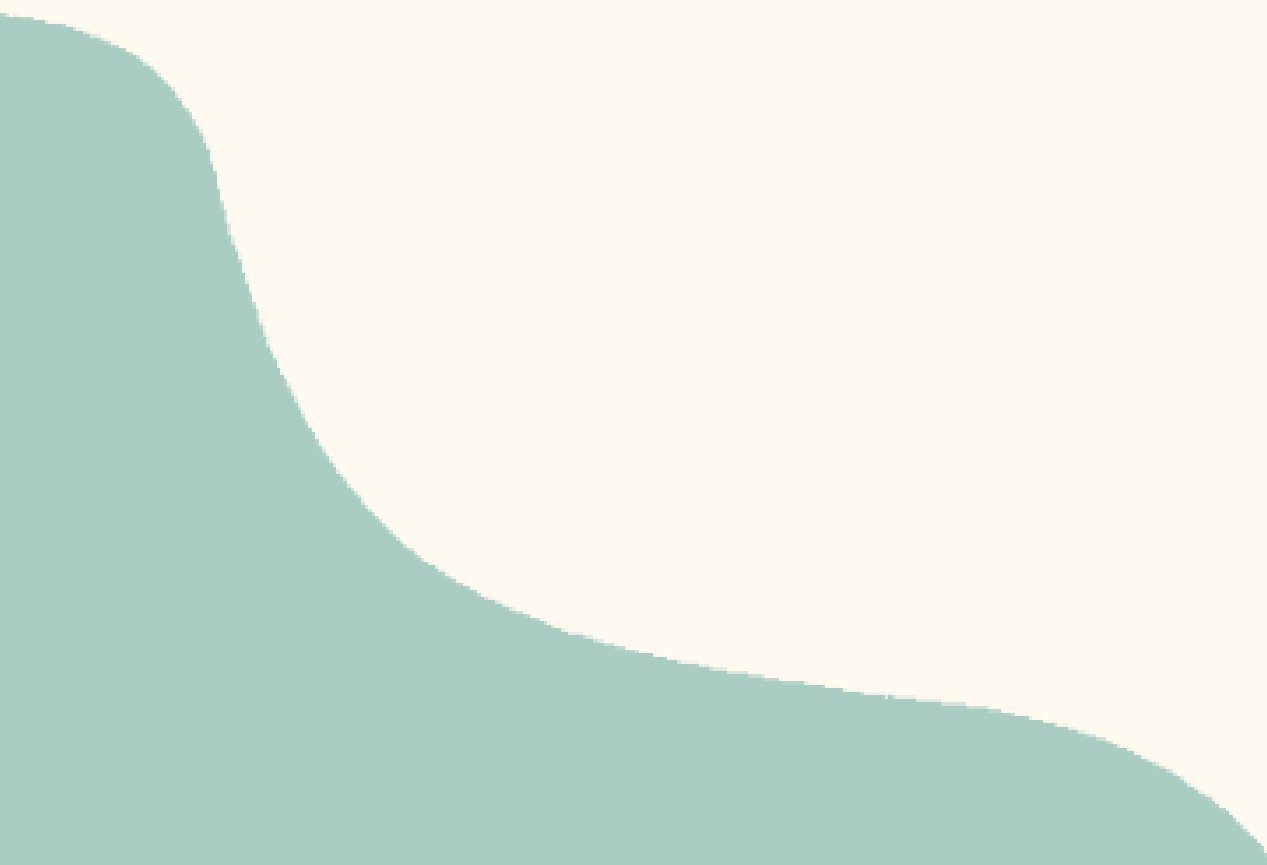
Alert và Dialog:

- 04**
- Hiển thị thông báo hoặc cảnh báo cho người dùng khi xảy ra lỗi hoặc cần xác nhận.





Kỹ thuật hướng đối tượng



01

Encapsulation (Đóng gói)

- Các lớp có thuộc tính riêng tư: Các thuộc tính trong các lớp (e.g., airplaneId, name, seatCapacity) được khai báo là private, chỉ có thể truy cập thông qua các getter và setter.
- Bảo vệ dữ liệu: Các setter kiểm tra và giới hạn giá trị đầu vào, đảm bảo dữ liệu luôn hợp lệ.

02

Inheritance (Kế thừa)

- Lớp cha Employee: Các lớp con như FlightCrew và GroundStaff kế thừa từ Employee, chia sẻ các thuộc tính và hành vi chung.
- Tái sử dụng mã nguồn (Code Reusability): Các lớp con không cần định nghĩa lại các thuộc tính và phương thức chung, như getId(), getName()...

Kỹ thuật hướng đối tượng

03

Polymorphism (Đa hình)

- Ghi đè phương thức (Method Overriding):

Các lớp con như FlightCrew và GroundStaff ghi đè phương thức getDetails() của lớp cha Employee để hiển thị thông tin chi tiết riêng.

- Đa hình qua tham chiếu lớp cha:

Các đối tượng FlightCrew và GroundStaff được xử lý thông qua tham chiếu kiểu Employee trong các TableView hoặc danh sách nhân viên.

04

Abstraction (Trừu tượng hóa)

- Ẩn chi tiết phức tạp:

Các dịch vụ (e.g., AirplaneService, FlightService) cung cấp giao diện đơn giản để tương tác với cơ sở dữ liệu, ẩn đi chi tiết truy vấn SQL.

- Tách biệt logic hiển thị và xử lý:

Các lớp Controller chịu trách nhiệm giao tiếp với giao diện người dùng (JavaFX), trong khi các lớp Service xử lý nghiệp vụ và giao tiếp với cơ sở dữ liệu.



DEMO



Xin cảm ơn!

Các bạn có bất kỳ câu hỏi nào cho chúng tôi không?