**ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**

**Báo cáo Project**

**Học phần: Lập trình Hướng đối tượng**

**Đề tài: Quản lý chuyến bay**

Mã lớp học: 151964

Giảng viên hướng dẫn: TS. Lê Đức Hậu

Nhóm sinh viên thực hiện: Hoàng Yến Nhi – 20225898

Nguyễn Khánh Duy – 20225830

Nguyễn Văn Phú – 20225658

Trần Lê Anh Minh – 20225652

Đỗ Tiến Đạt – 20225700

*Hà Nội, tháng 12 năm 2024*

**Mục lục**

**PHẦN 1: MÔ TẢ DỰ ÁN**

1. Mô tả nghiệp vụ
2. Các chức năng tiêu biểu

**PHẦN 2: THIẾT KẾ CƠ SỞ DỮ LIỆU**

1. Danh sách các bảng
2. Mô tả các bảng và quan hệ của chúng

**PHẦN 3: BIỂU ĐỒ UML**

1. Biểu đồ Class
2. Biểu đồ UseCase

**PHẦN 4: KỸ THUẬT VÀ CÔNG NGHỆ ĐÃ ÁP DỤNG**

1. Kỹ thuật hướng đối tượng đã sử dụng
2. Encapsulation (Đóng gói)
3. Inheritance (Kế thừa)
4. Polymorphism (Đa hình)
5. Abstraction (Trừu tượng hóa)
6. Các kỹ thuật khác
7. Association (Kết hợp)
8. Composition (Thành phần)
9. Data Binding (Ràng buộc dữ liệu)
10. Error Handling (Xử lý lỗi)
11. Dependency Injection (Tiêm phụ thuộc)
12. Persistence (Lưu trữ)
13. Công nghệ đã áp dụng
14. Java
15. MySQL
16. JDBC (Java Database Connectivity)
17. JavaFX Properties
18. JSON
19. JavaFX FXML

# PHẦN 1: 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.

1. **Mô tả nghiệp vụ**  
    Hệ thống lưu trữ và quản lý thông tin chuyến bay (***Flight***), bao gồm mã chuyến bay (***flightNumber***),địa điểm khởi hành và địa điểm cất cánh (***departureLocation, arrivalLocation***), thời gian khởi hành và hạ cánh (***departureTime, arrivalTime***), trạng thái (***status***), máy bay được gán (***airplaneId***) và cổng khởi hành (***gateNumber***), giúp điều phối chuyến bay hiệu quả.   
    Nghiệp vụ xem thông tin vé (***Ticket***) và hành khách (***Passenger***) cho phép theo dõi chi tiết số lượng vé đã bán, thông tin hành khách như tên (***name***), email (***email***), số điện thoại (***phone***), và liên kết chúng với từng chuyến bay (***flightNumber***).   
    Hệ thống cũng quản lý trạng thái và sử dụng cổng sân bay (***Gate***), đảm bảo tối ưu hóa tài nguyên với các trạng thái khả dụng (***isAvailable***).   
    Ngoài ra, chức năng quản lý nhân viên (***Employee***), bao gồm phi hành đoàn (***FlightCrew***) và nhân viên mặt đất (***GroundStaff***), giúp phân công nhiệm vụ (***assignmentDate***) chính xác và theo dõi tình trạng công việc.   
    Lịch sử chuyến bay (***FlightHistory***) được lưu trữ đầy đủ thông tin, bao gồm mã chuyến bay, thời gian, trạng thái, danh sách vé (***tickets***), hành khách và nhân viên liên quan.   
    Giao diện người dùng được thiết kế trực quan, cho phép thao tác thêm, sửa, xóa và xem thông tin một cách dễ dàng. Hệ thống hướng đến mục tiêu tối ưu hóa vận hành, giảm thiểu lỗi và hỗ trợ ra quyết định nhanh chóng.

### Các chức năng tiêu biểu

#### 1. Đối với chuyến bay (Flight):

* **Thêm thông tin mới**:
  + Thêm chuyến bay, cổng sân bay, phi hành đoàn, nhân viên mặt đất.
* **Xem thông tin chi tiết**:
  + Xem danh sách chuyến bay, hành khách, phi hành đoàn, nhân viên mặt đất, và vé.
  + Xem danh sách lịch sử chuyến bay để phân tích và báo cáo.
* **Cập nhật trạng thái và thông tin**:
  + Thay đổi trạng thái chuyến bay (Scheduled, Delayed, Cancelled, Landed).
  + Cập nhật thông tin chuyến bay như thời gian khởi hành/hạ cánh, cổng khởi hành, và máy bay được sử dụng.
* **Tính toán và kiểm tra**:
  + Tính tổng số vé và tổng số hành khách của từng chuyến bay.
  + Kiểm tra tính khả dụng của cổng sân bay để tránh xung đột.

#### 2. Đối với hành khách (Passenger):

* **Xem thông tin cá nhân và vé**:
  + Kiểm tra thông tin cá nhân như tên, email, số điện thoại.
  + Xem thông tin chuyến bay và ghế ngồi.

#### 3. Đối với phi hành đoàn (Flight Crew):

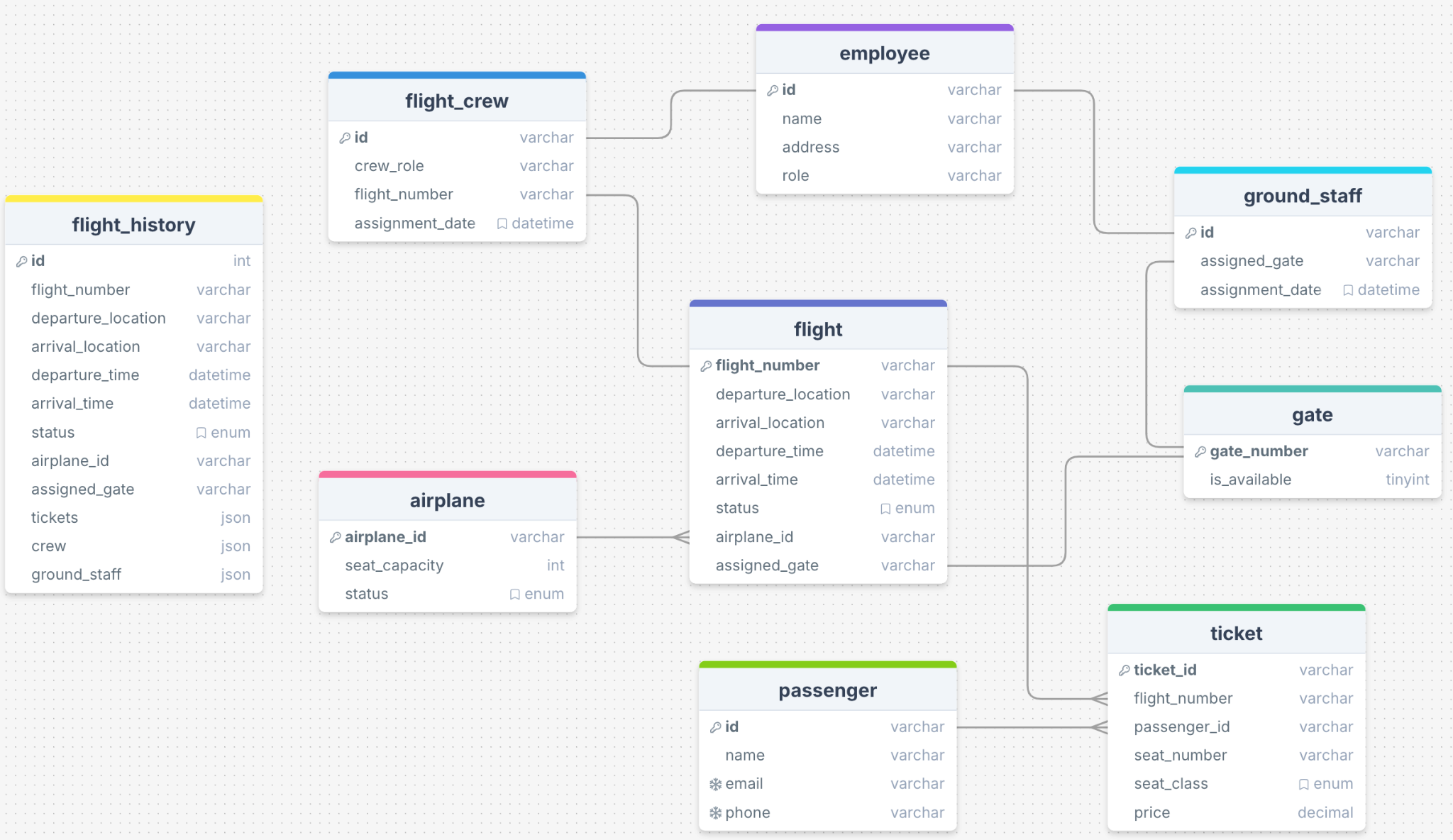
* **Xem nhiệm vụ**:
  + Xem danh sách các chuyến bay được phân công.
  + Kiểm tra chi tiết vai trò (Crew Role) và nhiệm vụ trong từng chuyến bay.

#### 4. Đối với nhân viên mặt đất (Ground Staff):

* **Xem nhiệm vụ**:
  + Kiểm tra danh sách các chuyến bay được phân công.
  + Xem thời gian và nhiệm vụ liên quan đến việc hỗ trợ các chuyến bay.

**PHẦN 2: THIẾT KẾ CƠ SỞ DỮ LIỆU**

### Danh sách các bảng



### Mô tả các bảng và quan hệ của chúng

### Cơ sở dữ liệu bao gồm các bảng sau:

#### 1. Bảng employee

* Thuộc tính:
  + id (PK): Mã định danh của nhân viên.
  + name: Tên nhân viên.
  + address: Địa chỉ của nhân viên.
  + role: Vai trò của nhân viên (ví dụ: phi hành đoàn, nhân viên mặt đất).
* Quan hệ:
  + Liên kết với bảng flight\_crew và ground\_staff để xác định vai trò của nhân viên.

#### 2. Bảng flight\_crew

* Thuộc tính:
  + id (PK, FK): Liên kết đến employee.id để xác định nhân viên thuộc phi hành đoàn.
  + crew\_role: Vai trò của nhân viên trong phi hành đoàn (ví dụ: phi công, tiếp viên).
  + flight\_number (FK): Mã chuyến bay mà phi hành đoàn được gán.
  + assignment\_date: Ngày được phân công làm nhiệm vụ.
* Quan hệ:
  + Liên kết với bảng flight qua flight\_number để gán nhiệm vụ phi hành đoàn.

#### 3. Bảng ground\_staff

* Thuộc tính:
  + id (PK, FK): Liên kết đến employee.id để xác định nhân viên thuộc nhân viên mặt đất.
  + assigned\_gate: Mã cổng sân bay mà nhân viên được phân công.
  + assignment\_date: Ngày được giao nhiệm vụ.
* Quan hệ:
  + Liên kết với bảng gate qua assigned\_gate để phân công nhân viên mặt đất tại cổng.

#### 4. Bảng flight

* Thuộc tính:
  + flight\_number (PK): Mã định danh của chuyến bay.
  + departure\_location: Địa điểm khởi hành.
  + arrival\_location: Địa điểm đến.
  + departure\_time: Thời gian khởi hành.
  + arrival\_time: Thời gian hạ cánh.
  + status: Trạng thái chuyến bay (Scheduled, Delayed, Cancelled, Landed).
  + airplane\_id (FK): Mã máy bay được gán cho chuyến bay.
  + assigned\_gate (FK): Cổng được gán cho chuyến bay.
* Quan hệ:
  + Liên kết với bảng airplane qua airplane\_id để sử dụng máy bay.
  + Liên kết với bảng gate qua assigned\_gate để gán cổng khởi hành.
  + Liên kết với bảng flight\_crew và ground\_staff để quản lý nhân sự liên quan.

#### 5. Bảng airplane

* Thuộc tính:
  + airplane\_id (PK): Mã định danh máy bay.
  + seat\_capacity: Số ghế của máy bay.
  + status: Tình trạng của máy bay (Available, In Maintenance, In Flight).
* Quan hệ:
  + Liên kết với bảng flight qua airplane\_id.

#### 6. Bảng gate

* Thuộc tính:
  + gate\_number (PK): Mã định danh cổng sân bay.
  + is\_available: Trạng thái khả dụng của cổng (0: không khả dụng, 1: khả dụng).
* Quan hệ:
  + Liên kết với bảng flight qua assigned\_gate.
  + Liên kết với bảng ground\_staff qua assigned\_gate.

#### 7. Bảng passenger

* Thuộc tính:
  + id (PK): Mã định danh hành khách.
  + name: Tên hành khách.
  + email: Địa chỉ email của hành khách (duy nhất).
  + phone: Số điện thoại của hành khách (duy nhất).
* Quan hệ:
  + Liên kết với bảng ticket qua id để xác định vé của hành khách.

#### 8. Bảng ticket

* Thuộc tính:
  + ticket\_id (PK): Mã định danh vé.
  + flight\_number (FK): Liên kết đến flight\_number trong bảng flight.
  + passenger\_id (FK): Liên kết đến id trong bảng passenger.
  + seat\_number: Số ghế của vé.
  + seat\_class: Hạng ghế (Economy, Business).
  + price: Giá vé.
* Quan hệ:
  + Liên kết với bảng flight và passenger để quản lý vé.

#### 9. Bảng flight\_history

* Thuộc tính:
  + id (PK): Mã định danh lịch sử chuyến bay.
  + flight\_number: Mã chuyến bay (chỉ lưu trữ, không liên kết trực tiếp).
  + departure\_location: Địa điểm khởi hành.
  + arrival\_location: Địa điểm đến.
  + departure\_time: Thời gian khởi hành.
  + arrival\_time: Thời gian hạ cánh.
  + status: Trạng thái chuyến bay (Cancelled, Landed).
  + airplane\_id: Mã máy bay sử dụng trong chuyến bay.
  + assigned\_gate: Mã cổng được gán cho chuyến bay.
  + tickets: Danh sách vé dưới dạng JSON.
  + crew: Danh sách phi hành đoàn dưới dạng JSON.
  + ground\_staff: Danh sách nhân viên mặt đất dưới dạng JSON.
* Quan hệ:
  + Không liên kết trực tiếp với các bảng khác, chỉ lưu trữ dữ liệu lịch sử dưới dạng độc lập để phục vụ tra cứu và báo cáo.

### Ghi chú về flight\_history:

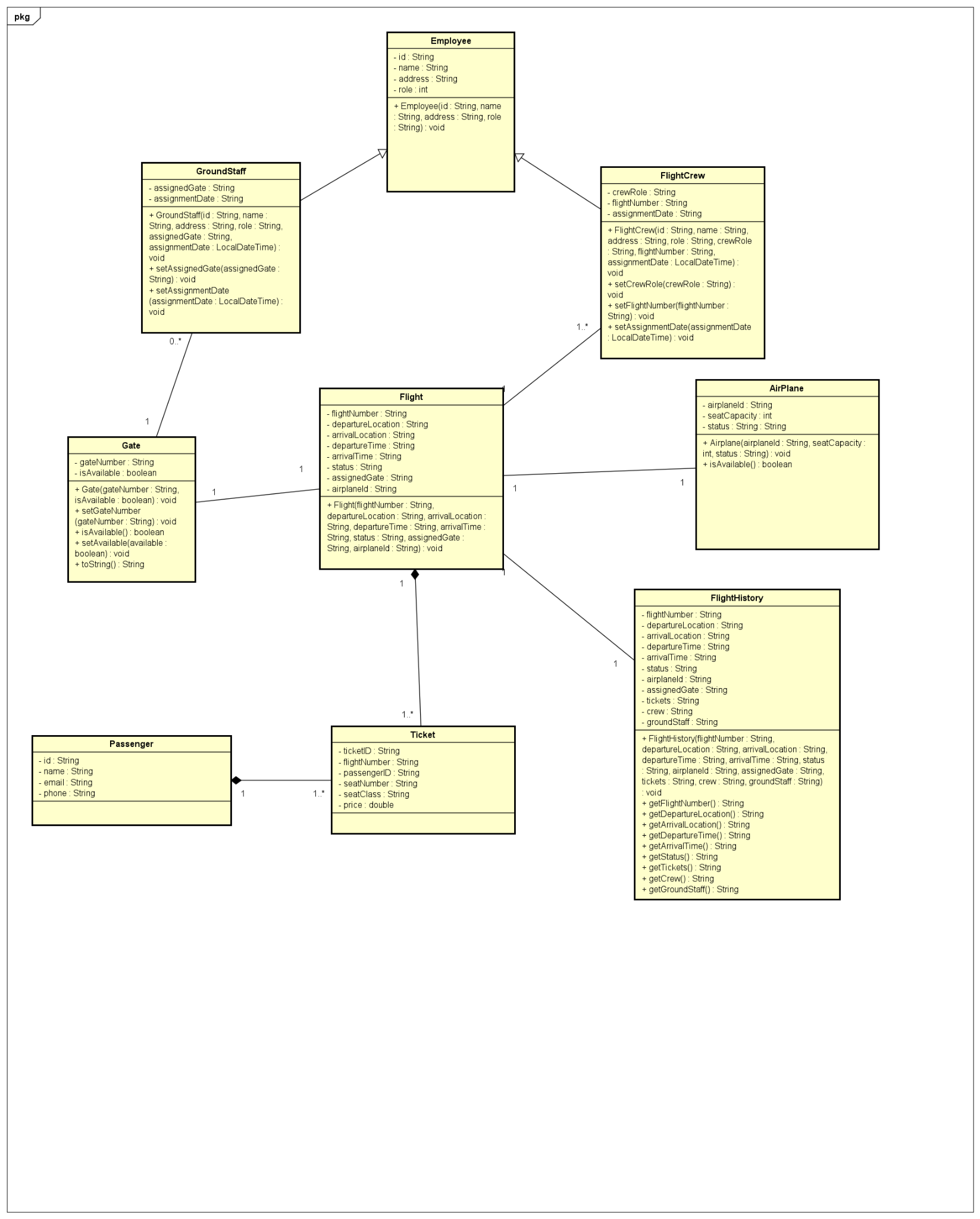
Bảng này hoạt động như một bảng lưu trữ dữ liệu snapshot (bản chụp nhanh) của thông tin chuyến bay khi chuyến bay hoàn thành hoặc bị hủy. Mặc dù không có quan hệ khóa ngoại, dữ liệu trong bảng có thể phản ánh các thông tin từ các bảng khác vào thời điểm lưu trữ lịch sử.

### Các liên kết giữa các bảng:

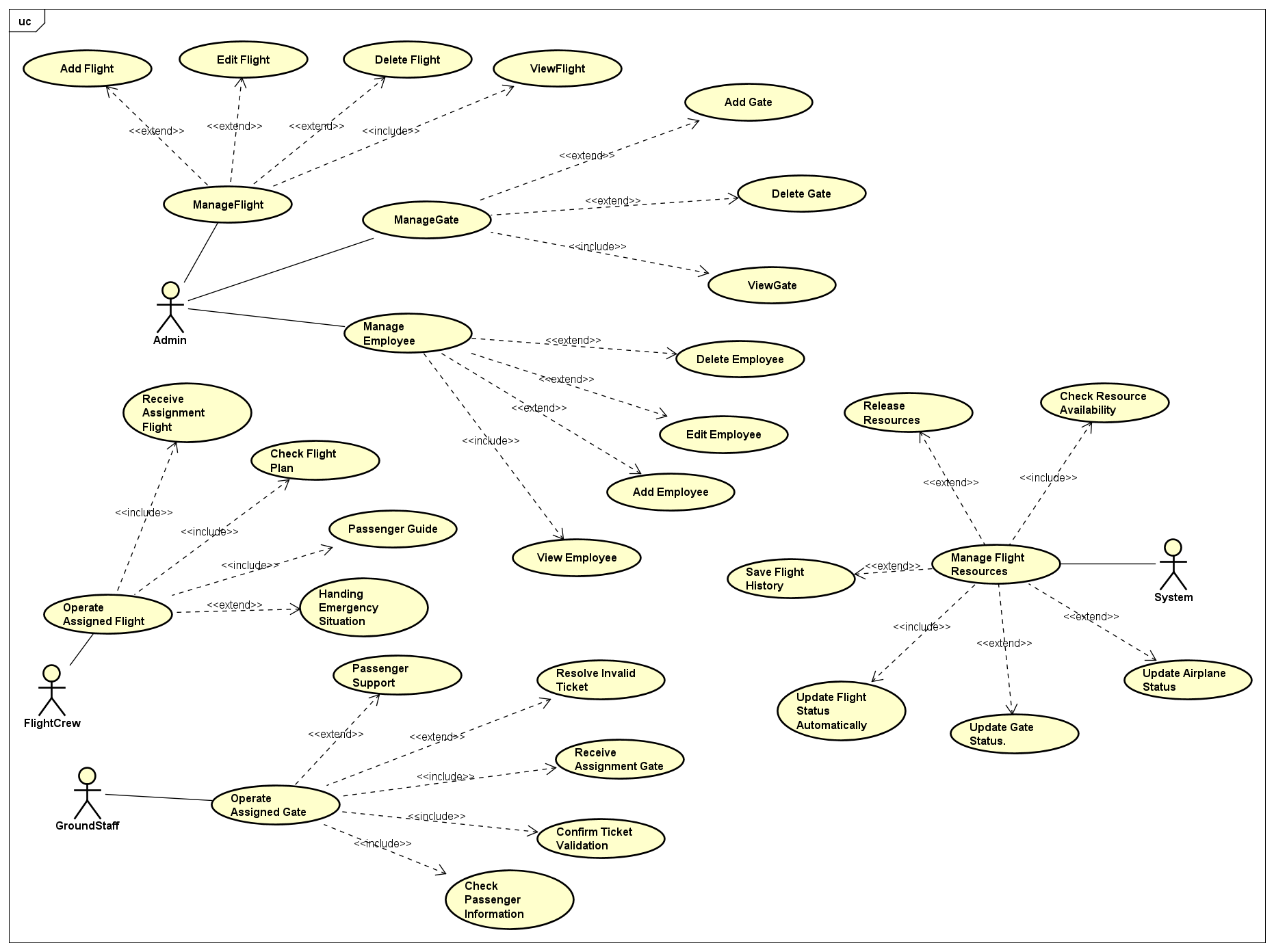
* Liên kết giữa employee và flight\_crew: 1-1
* Liên kết giữa employee và ground\_staff: 1-1
* Liên kết giữa flight và airplane: 1-1
* Liên kết giữa flight và gate: 1-1
* Liên kết giữa flight và flight\_crew: 1-n
* Liên kết giữa ground\_staff và gate: n-1
* Liên kết giữa flight và ticket: 1-n
* Liên kết giữa ticket và passenger: 1-1

**PHẦN 3: BIỂU ĐỒ UML**

### Biểu đồ Class

****

### Biểu đồ UseCase

****

**PHẦN 4: KỸ THUẬT VÀ CÔNG NGHỆ ĐÃ ÁP DỤNG**

**I. Kỹ thuật hướng đối tượng đã sử dụng**

1. 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.

2. 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 (e.g., id, name, address, role).
* Lớp con mở rộng thêm thuộc tính riêng
* 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().

3. 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.

4. 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.

**II. Các kỹ thuật khác**

1. Association (Kết hợp)

* **Mối quan hệ giữa các đối tượng**:
* Flight chứa thông tin về Gate, Airplane, và FlightCrew.
* Mối quan hệ được thể hiện qua thuộc tính trong các lớp

2. Composition (Thành phần)

* **Lớp TicketAdapter**:

Kết hợp dữ liệu từ Ticket và thông tin hành khách để tạo ra một đối tượng mới phục vụ việc hiển thị trong bảng.

3. Data Binding (Ràng buộc dữ liệu)

* **Sử dụng JavaFX Property**:

Các thuộc tính như StringProperty và BooleanProperty được sử dụng để ràng buộc dữ liệu giữa mô hình và giao diện người dùng.

4. Error Handling (Xử lý lỗi)

* **Kiểm tra và xử lý lỗi**:

Các lớp dịch vụ sử dụng khối try-catch để bắt và xử lý lỗi khi truy vấn cơ sở dữ liệu.

5. Dependency Injection (Tiêm phụ thuộc)

* **Quản lý phụ thuộc giữa các lớp**:

Các lớp Controller sử dụng các lớp Service để thực hiện logic nghiệp vụ, không trực tiếp xử lý cơ sở dữ liệu.

6. Persistence (Lưu trữ)

* **Lưu trữ dữ liệu**:

Các lớp như FlightHistory sử dụng JSON để lưu trữ chi tiết về vé, phi hành đoàn, và nhân viên mặt đất trong cơ sở dữ liệu.

**III. Công nghệ đã áp dụng**

1. Java

* **Ngôn ngữ lập trình chính**:
* Dùng để xây dựng toàn bộ hệ thống, từ logic nghiệp vụ đến giao diện người dùng.
* **JavaFX**:
* Framework cho giao diện người dùng (UI), cung cấp các thành phần như TableView, ComboBox, Button, Label, v.v.
* Hỗ trợ ràng buộc dữ liệu (Property Binding) giữa mô hình (Model) và giao diện.

2. MySQL

* **Hệ quản trị cơ sở dữ liệu (DBMS)**:
* Lưu trữ thông tin về chuyến bay, máy bay, nhân viên, hành khách, vé, và lịch sử chuyến bay.
* **Truy vấn SQL**:
* Các lớp dịch vụ (e.g., FlightService, PassengerService) thực hiện CRUD thông qua JDBC với các câu lệnh SQL.

3. JDBC (Java Database Connectivity)

* **Kết nối Java với MySQL**:
* Lớp DatabaseConnection sử dụng JDBC để tạo và quản lý kết nối đến cơ sở dữ liệu.

4. JavaFX Properties

* **Ràng buộc dữ liệu**:
* Sử dụng các lớp như StringProperty, BooleanProperty, và DoubleProperty để kết nối dữ liệu giữa mô hình và giao diện người dùng.
* Tích hợp chặt chẽ với các thành phần JavaFX như TableView.

5. JSON

* **Lưu trữ dữ liệu phức tạp**:
* Sử dụng JSON để lưu danh sách vé, phi hành đoàn, và nhân viên mặt đất trong bảng flight\_history.
* Tương tác JSON thông qua các thư viện như org.json.

6. JavaFX FXML

* **Mô tả giao diện người dùng**:
* Các tệp .fxml được sử dụng để định nghĩa giao diện, giúp tách biệt thiết kế giao diện và logic xử lý.
* Kết nối với các lớp Controller để xử lý sự kiện và quản lý dữ liệu.