ĐẠI HỌC QUỐC GIA HÀ NỘI TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

Nhóm: ĐÂY LÀ NHÓM SQL

- *Nguyễn Phương Nam 23020406*
- *Nguyễn Đình Quyền 23020422*
- *Trần Doãn Thắng 23020438*
- *Nguyễn Trọng Hồng Phúc 23020410*

HỆ THỐNG QUẢN LÝ ĐẶT VÉ TÀU

BÀI TẬP LỚN MÔN CƠ SỞ DỮ LIỆU
Ngành: Trí tuệ nhân tạo

HÀ NỘI - 2024

ĐẠI HỌC QUỐC GIA HÀ NỘI

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

BÁO CÁO BÀI TẬP LỚN

Học phần: Cơ sở dữ liệu

ĐỀ TÀI: CƠ SỞ DỮ LIỆU CHO HỆ THỐNG ĐẶT VÉ TÀU

Nhóm thực hiện:

Nguyễn Phương Nam	23020406
Nguyễn Đình Quyền	23020422
Trần Doãn Thắng	23020438
Nguyễn Trọng Hồng Phúc	23020410

Github Repo: github

I.PHÁT BIỂU BÀI TOÁN

1. Ý nghĩa

Hệ thống quản lý vé tàu:

Hệ thống này được thiết kế để quản lý việc đặt vé tàu, quản lý thông tin tài khoản người dùng, thông tin hành khách, lịch trình tàu, và các điểm dừng. Mục tiêu chính là cung cấp một nền tảng dễ sử dụng để người dùng có thể đặt vé trực tuyến, kiểm tra thông tin chuyến đi và quản lý thông tin cá nhân.

2. Các thực thể (Entities) và chức năng

Dựa trên hệ thống bán vé tàu, ta có các thực thể chính sau:

a. Tài khoản (Account):

- **Chức năng:** Lưu trữ thông tin đăng nhập của người dùng để truy cập và quản lý thông tin cá nhân, đặt vé.
- **Thông tin cần thiết:**
  - Username: Tên đăng nhập.
  - Password: Mật khẩu.
  - Email_Id: Địa chỉ email để liên lạc.
  - Address: Địa chỉ nhà (tùy chọn).

b. Tàu (Train):

- **Chức năng:** Quản lý thông tin chi tiết của các tàu hoạt động, bao gồm tên tàu, số lượng ghế trong từng loại khoang, và tiện ích bổ sung.
- **Thông tin cần thiết:**
  - Train_No: Mã tàu.
  - Name: Tên tàu.
  - Seat_Sleeper: Số ghế giường nằm.
  - Seat_First_Class_AC: Số ghế hạng nhất có điều hòa.
  - Seat_Second_Class_AC: Số ghế hạng hai có điều hòa.
  - Seat_Third_Class_AC: Số ghế hạng ba có điều hòa.
  - Wifi: Có cung cấp wifi (Y/N).
  - Food: Có cung cấp thức ăn (Y/N).
  - Run_On_*: Chạy vào các ngày trong tuần.

c. Ga tàu (Station):

- **Chức năng:** Quản lý danh sách các ga tàu, hỗ trợ cho việc lên lịch trình dừng đỗ của tàu.
- **Thông tin cần thiết:**

- Station_Code: Mã ga.
- Station_Name: Tên ga.

d. Vé tàu (Ticket):

- **Chức năng:** Lưu trữ thông tin về vé tàu, bao gồm chi tiết toa, số ghế, và ngày khởi hành.
- **Thông tin cần thiết:**
  - Ticket_No: Mã vé.
  - Coach: Số toa.
  - Seat_no: Số ghế.
  - Train_No: Mã tàu (liên kết với tàu).
  - Date_of_Journey: Ngày khởi hành.
  - Username: Tên người dùng đặt vé (liên kết với tài khoản).

e. Điểm dừng (Stoppage):

- **Chức năng:** Quản lý thông tin các điểm dừng và thời gian đến/rời của mỗi tàu.
- **Thông tin cần thiết:**
  - Train_No: Mã tàu.
  - Station_Code: Mã ga dừng.
  - Arrival_Time: Thời gian tàu đến ga.
  - Departure_Time: Thời gian tàu rời ga.

f. Hành khách (Passenger):

- **Chức năng:** Lưu trữ thông tin chi tiết của từng hành khách liên quan đến vé tàu.
- **Thông tin cần thiết:**
  - Passenger_Id: Mã hành khách.
  - First_Name: Tên.
  - Last_Name: Họ.
  - Gender: Giới tính.
  - Phone_No: Số điện thoại.
  - Ticket_No: Mã vé (liên kết với vé tàu).
  - Age: Tuổi.
  - Class: Hạng ghế.

g. Liên hệ (Contact):

- **Chức năng:** Quản lý thông tin liên hệ bổ sung của người dùng.
- **Thông tin cần thiết:**
  - Username: Tên đăng nhập của người dùng (liên kết với tài khoản).
  - Phone_No: Số điện thoại liên hệ.

II. MÔ TẢ NGHIỆP VỤ HỆ THỐNG

1. **Người dùng đăng ký tài khoản:**
    - Người dùng nhập thông tin Username, Password, Email_Id, và (tùy chọn) Address để tạo tài khoản.
2. **Người dùng đăng nhập:**
    - Sử dụng Username và Password để truy cập hệ thống.
3. **Tìm kiếm tàu:**
    - Người dùng nhập điểm đi, điểm đến, và ngày khởi hành để tìm kiếm các tàu phù hợp.
4. **Hiển thị thông tin tàu:**
    - Hệ thống hiển thị danh sách tàu với các thông tin như Name, Seat_Sleeper, Seat_First_Class_AC, và tiện ích bổ sung như Wifi hoặc Food.
5. **Chọn vé:**
    - Người dùng chọn tàu, nhập thông tin hành khách (họ tên, tuổi, số điện thoại), và chọn ghế (hạng ghế, số ghế).
6. **Quản lý vé:**
    - Sau khi đặt vé, thông tin vé sẽ được lưu trữ trong thực thể **Ticket**. Người dùng có thể xem lại vé đã đặt thông qua hệ thống.
7. **Lịch trình tàu và điểm dừng:**
    - Thông tin lịch trình tàu được quản lý trong thực thể **Stoppage**, liên kết với các ga tàu trong **Station**.

III. MÔ HÌNH ER

1. **Mô hình thực thể/quan hệ (Entity/Relationship Model - ER Model):**
   -Mô hình thực thể/quan hệ (Entity/Relationship Model - ER Model) là một cách biểu diễn dữ liệu của hệ thống thông qua các thực thể (entities) và mối quan hệ (relationships) giữa các thực thể đó. Yêu cầu này liên quan đến việc xây dựng mô hình này để hiểu và tổ chức dữ liệu của hệ thống.
2. **Đề xuất mô hình thực thể cho hệ thống**

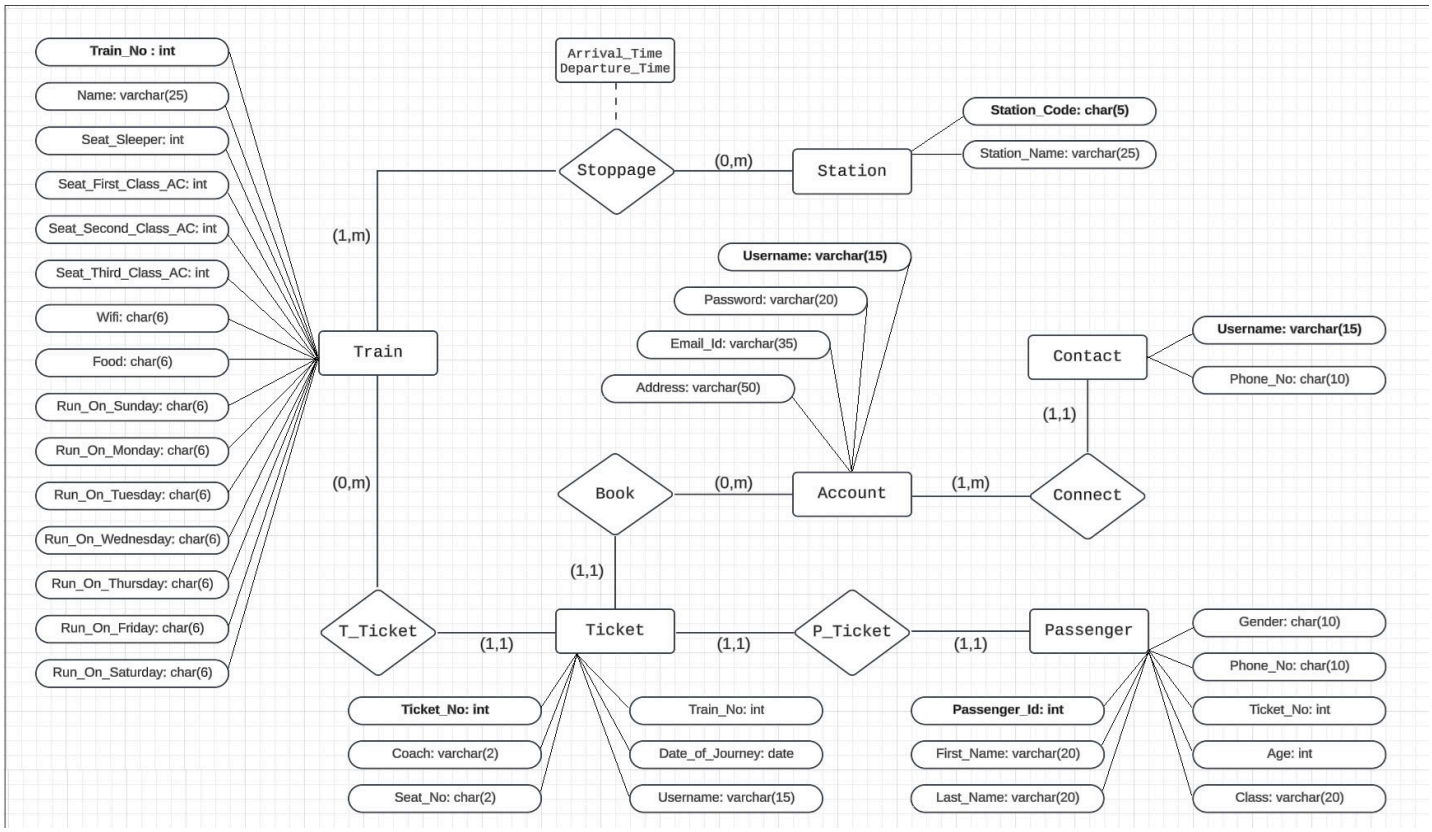
Mô hình ER cho hệ thống quản lý bán vé tàu

1. Thực thể (Entities) và thuộc tính (Attributes):

1. **Tài khoản (Account):**
   - **Attributes:**
     - **Username: Tên đăng nhập.**
     - **Password: Mật khẩu.**
     - **Email_Id: Địa chỉ email.**
     - **Address: Địa chỉ nhà (nếu có).**
2. **Tàu (Train):**
   - **Attributes:**
     - **Train_No: Mã tàu.**
     - **Name: Tên tàu.**
     - **Seat_Sleeper: Số ghế giường nằm.**
     - **Seat_First_Class_AC: Số ghế hạng nhất có điều hòa.**
     - **Seat_Second_Class_AC: Số ghế hạng hai có điều hòa.**
     - **Seat_Third_Class_AC: Số ghế hạng ba có điều hòa.**
     - **Wifi: Có wifi (Y/N).**
     - **Food: Có cung cấp thức ăn (Y/N).**
     - **Run_On_*: Chạy vào các ngày trong tuần.**
3. **Ga tàu (Station):**
   - **Attributes:**
     - **Station_Code: Mã ga.**
     - **Station_Name: Tên ga.**
4. **Vé tàu (Ticket):**
   - **Attributes:**
     - **Ticket_No: Mã vé.**
     - **Coach: Số toa**
     - **Seat_no: Số ghế**
     - **Train_No: Mã tàu.**

        - **Date_of_Journey: Ngày khởi hành.**
        - **Username: Tên người dùng đặt vé.**
5. **Điểm dừng (Stoppage):**
    - **Attributes:**
        - **Train_No: Mã tàu.**
        - **Station_Code: Mã ga dừng.**
        - **Arrival_Time: Thời gian tàu đến.**
        - **Departure_Time: Thời gian tàu rời ga.**
6. **Hành khách (Passenger):**
    - **Attributes:**
        - **Passenger_Id: Mã hành khách.**
        - **First_Name: Tên.**
        - **Last_Name: Họ.**
        - **Gender: Giới tính.**
        - **Phone_No: Số điện thoại.**
        - **Ticket_No: Mã vé.**
        - **Age: Tuổi.**
        - **Class: Hạng ghế.**
7. **Liên hệ (Contact):**
    - **Attributes:**
        - **Username: Tên đăng nhập của người dùng.**
        - **Phone_No: Số điện thoại liên hệ.**

Mô hình ER



Đặc trưng:

- **Hình chữ nhật:** Đại diện cho thực thể.
- **Hình elip:** Đại diện cho thuộc tính.
- **Hình thoi:** Đại diện cho mối quan hệ.
- **Đường thẳng:** Kết nối thực thể với thuộc tính hoặc mối quan hệ.

3. Mối quan hệ (Relationships):

- **(1,1):** Một thực thể chỉ có thể liên kết với một thực thể duy nhất của loại khác.
- **(1,n):** Một thực thể có thể liên kết với một hoặc nhiều thực thể của loại khác.
- **(0,n):** Một thực thể có thể không liên kết với bất kỳ thực thể nào hoặc có thể liên kết với nhiều thực thể của loại khác.

a. Người dùng đặt vé tàu:

- **Tài khoản (Account) → Đặt (Books) → Vé tàu (Ticket)**
  - **Mô tả: Một người dùng có thể đặt nhiều vé tàu thông qua tài khoản của họ.**
  - **Quan hệ:**
    - **Một tài khoản có thể không đặt hoặc đặt nhiều vé (0, m).**
    - **Một vé chỉ được đặt bởi một tài khoản duy nhất (1, 1).**

b. Tàu dừng tại ga:

- **Tàu (Train) → Dừng tại (Stoppage) → Ga tàu (Station)**
  - **Mô tả: Một tàu có thể dừng tại nhiều ga, và một ga có thể được nhiều tàu ghé qua.**
  - **Quan hệ:**
    - **Một tàu có thể dừng tại nhiều ga (1, m).**
    - **Một ga có thể không có tàu ghé hoặc được nhiều tàu ghé qua (0, m).**

c. Vé gắn với hành khách:

- **Vé tàu (Ticket) → Thuộc về (P_Ticket) → Hành khách (Passenger)**
  - **Mô tả: Một vé tàu có thể gắn với nhiều hành khách (nhóm người đi cùng). Mỗi hành khách sẽ có thông tin riêng như tên, tuổi, giới tính và ghế ngồi trên vé.**
  - **Quan hệ:**
    - **Một vé có thể gắn với một hành khách (1, 1).**
    - **Một hành khách chỉ gắn với một vé duy nhất (1, 1).**

d. Vé gắn với tàu:

- **Vé tàu (Ticket) → Thuộc về (T_Ticket) → Tàu (Train)**
  - **Mô tả: Mỗi vé sẽ gắn với một tàu cụ thể để xác định chuyến đi. Số lượng ghế còn trống và loại ghế (Sleeper, First_Class_AC, etc.) trên tàu sẽ được quản lý dựa trên vé được đặt hoặc hủy.**
  - **Quan hệ:**
    - **Một tàu có thể không có hoặc có nhiều vé được đặt (0, m).**
    - **Một vé chỉ thuộc về một tàu duy nhất (1, 1).**

e. Thông tin liên lạc của tài khoản:

- **Tài khoản (Account) → Liên lạc (Contact)**
    - **Mô tả: Một tài khoản có thể có nhiều số điện thoại được lưu trong hệ thống.**
    - **Quan hệ:**
        - **Một tài khoản có thể có nhiều số liên lạc (1, m).**
        - **Mỗi số liên lạc chỉ gắn với một tài khoản duy nhất (1, 1).**

4. Kết luận

Các mối quan hệ trên cho phép hệ thống quản lý mọi khía cạnh của việc đặt vé, bao gồm:

- Quản lý hành khách, vé, và tàu liên kết chặt chẽ với nhau.
- Quản lý thông tin người dùng và liên lạc của họ.
- Dễ dàng cập nhật số lượng ghế trống khi có thay đổi (đặt hoặc hủy vé).

IV. MÔ HÌNH QUAN HỆ

Chuyển đổi mô hình ER thành mô hình quan hệ nghĩa là chúng ta sẽ chuyển các thực thể (entities), thuộc tính (attributes), và các mối quan hệ (relationships) từ sơ đồ ER thành các bảng trong cơ sở dữ liệu.

Mục tiêu:

- **Biểu diễn dưới dạng mô hình quan hệ**: Tạo ra các bảng tương ứng cho từng thực thể và mối quan hệ.
- **Xác định phụ thuộc hàm**: Phân tích và chỉ ra các thuộc tính phụ thuộc vào khóa chính.
- **Chuẩn hóa cơ sở dữ liệu đến 3NF**: Đảm bảo không có dư thừa dữ liệu bằng cách loại bỏ các phụ thuộc bắc cầu (transitive dependency) và các thuộc tính không phụ thuộc hoàn toàn vào khóa chính.
- **Chuẩn hóa phụ thuộc hàm**:
    - 1NF: thuộc tính không có tập hợp
    - 2NF: 1NF + mỗi thuộc tính ngoài khóa chính phụ thuộc vào tất cả thuộc tính thuộc khóa chính
    - 3NF: 2NF + không có phụ thuộc bắc cầu giữa các thuộc tính không thuộc khóa
    - BCNF: 3NF + mọi phụ thuộc hàm đều có vế trái là siêu khóa
        - Siêu khóa: Giá trị của siêu khóa phải đảm bảo không có hai bộ nào trong bảng có giá trị giống nhau ở các thuộc tính của siêu khóa.

Giải quyết:

1. **Train**
   Train_No → (Name, Seat_Sleeper, Seat_First_Class_AC, Seat_Second_Class_AC, Seat_Third_Class_AC, Wifi, Food, Run_On_Sunday, Run_On_Monday, Run_On_Tuesday, Run_On_Wednesday, Run_On_Thursday, Run_On_Friday, Run_On_Saturday)

    - **Train_No là khóa chính của bảng TRAIN.**

- **Tất cả các thuộc tính khác trong bảng (Name, Seat_Sleeper, Seat_First_Class_AC, Seat_Second_Class_AC, Seat_Third_Class_AC, Wifi, Food, các ngày chạy tàu) đều phụ thuộc vào Train_No.**
- **Điều này có nghĩa là khi ta biết Train_No, ta có thể xác định tất cả các thông tin về tàu như tên tàu, số ghế của từng loại, tiện ích như wifi, food, và các ngày tàu hoạt động trong tuần.**

2. **Stoppage**
   (Train_No, Station_Code) → (Arrival_Time, Departure_Time)

   - **Train_No và Station_Code tạo thành khóa chính của bảng STOPPAGE.**
   - **Các thuộc tính Arrival_Time (giờ đến) và Departure_Time (giờ rời) phụ thuộc vào sự kết hợp của Train_No và Station_Code.**
   - **Điều này có nghĩa là khi ta biết tàu (Train_No) và ga (Station_Code), ta có thể xác định thời gian tàu đến và rời khỏi ga đó.**

3. **Ticket**

   Ticket_No → (Train_No, Date_of_Journey, Username)

   - **Ticket_No là khóa chính của bảng TICKET.**
   - **Các thuộc tính Train_No, Date_of_Journey (ngày đi) và Username (tài khoản người đặt vé) đều phụ thuộc vào Ticket_No.**
   - **Điều này có nghĩa là khi ta biết số vé (Ticket_No), ta có thể xác định tàu nào, ngày đi và tài khoản người đã đặt vé.**

4. **Station**

   Station_Code → Station_Name

   - **Station_Code là khóa chính của bảng STATION.**
   - **Thuộc tính Station_Name (tên ga) phụ thuộc vào Station_Code.**
   - **Điều này có nghĩa là khi ta biết mã ga (Station_Code), ta có thể biết tên ga tương ứng.**

5. **Passenger**

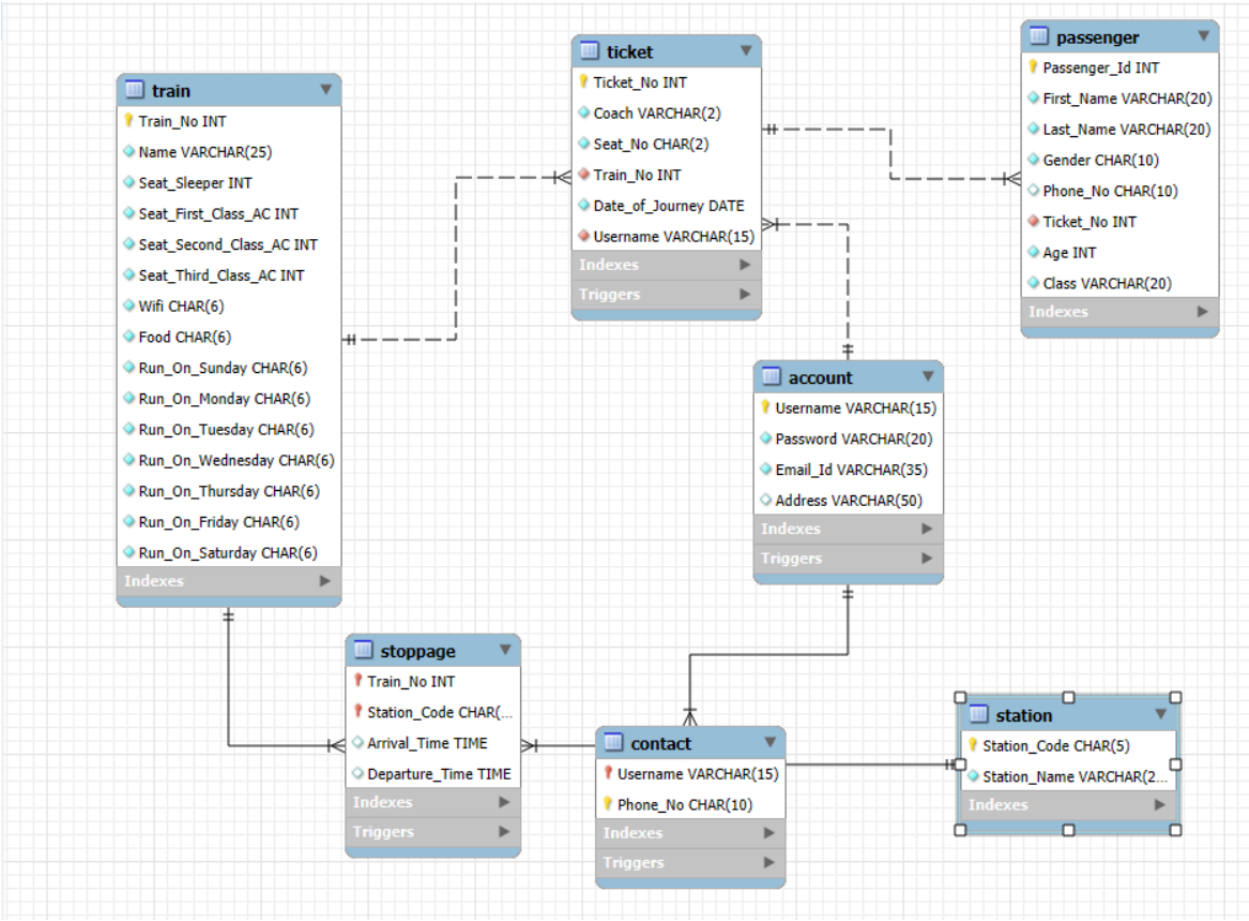
   Passenger_Id → (First_Name, Last_Name, Gender, Phone_No, Ticket_No, Age, Class)

- **Passenger_Id là khóa chính của bảng PASSENGER.**
- **Các thuộc tính như First_Name, Last_Name, Gender, Phone_No, Ticket_No, Age, và Class đều phụ thuộc vào Passenger_Id.**
- **Điều này có nghĩa là khi ta biết Passenger_Id, ta có thể xác định tất cả thông tin về hành khách như tên, giới tính, số điện thoại, vé tàu, tuổi và hạng vé của hành khách.**

6. **Account**

Username → (Password, Email_Id, Address)

- **Username là khóa chính của bảng ACCOUNT.**
- **Các thuộc tính Password, Email_Id và Address đều phụ thuộc vào Username.**
- **Điều này có nghĩa là khi ta biết tài khoản người dùng (Username), ta có thể biết mật khẩu, email và địa chỉ của người đó.**



Giải thích

- **First Normal Form (1NF - Chuẩn hóa dạng 1):** Theo quy tắc của Chuẩn hóa dạng 1, mỗi thuộc tính (cột) trong bảng phải chứa các giá trị nguyên tử (atomic values), nghĩa là mỗi cột không thể chứa nhiều giá trị (không được phép có giá trị đa trị).
    - Schema trên đã tuân thủ chuẩn 1NF vì tất cả các thuộc tính đều là nguyên tử, không có giá trị đa trị.
    - Tuy nhiên, trong trường hợp hành khách có thể có nhiều số điện thoại, điều này sẽ vi phạm quy tắc của 1NF nếu lưu trữ trực tiếp trong bảng Passenger. Do đó, bảng Contact được tạo ra để lưu trữ thông tin số điện thoại của hành khách, giúp duy trì chuẩn 1NF.
- **Second Normal Form (2NF - Chuẩn hóa dạng 2):**

  Một bảng được coi là ở Chuẩn hóa dạng 2 nếu thỏa mãn cả hai điều kiện sau:

+ Bảng phải ở 1NF (Chuẩn hóa dạng 1).
+ Không có thuộc tính không phải khóa phụ thuộc vào một phần của bất kỳ khóa ứng viên nào.
    - Trong bảng Passenger, nếu ta coi Ticket_No và First_Name là khóa ứng viên, thì thông tin Date_of_Birth (ngày sinh) sẽ phụ thuộc vào tên (First_Name), điều này sẽ vi phạm 2NF. Để khắc phục, ta cần đảm bảo rằng mỗi thuộc tính không phải khóa phải phụ thuộc vào toàn bộ khóa ứng viên, không chỉ một phần của nó.

- **Third Normal Form (3NF - Chuẩn hóa dạng 3):**

Một bảng đạt Chuẩn hóa dạng 3 nếu thỏa mãn cả hai điều kiện sau:

+ Bảng phải ở 2NF (Chuẩn hóa dạng 2).
+ Các phụ thuộc hàm chuyển tiếp của thuộc tính không phải khóa vào bất kỳ khóa siêu nào phải được loại bỏ.
    - Schema trên đã loại bỏ được các phụ thuộc hàm chuyển tiếp, ví dụ, thông tin về địa chỉ của người dùng không phụ thuộc vào bất kỳ thuộc tính không phải khóa nào mà chỉ phụ thuộc vào khóa chính (tên người dùng). Điều này đảm bảo rằng bảng đạt chuẩn 3NF.

Như vậy, các bảng này đã được chuẩn hóa lên đến 3NF, đảm bảo tính nhất quán và giảm thiểu sự dư thừa dữ liệu.

V. BẢNG ĐẶC TẢ YÊU CẦU DỮ LIỆU

TABLE	DATA ELEMENT	DESCRIPTION	COMPOSITION OR DATA TYPE	LENGTH	VALUES
Account	Username	Tên đăng nhập	varchar	15	NOT NULL, PRIMARY KEY
	Password	Mật khẩu	varchar	20	NOT NULL
	Email_Id	Địa chỉ email	varchar	35	NOT NULL
	Address	Địa chỉ nhà riêng	varchar	50	DEFAULT NULL
Train	Train_No	Mã tàu	int	x	NOT NULL, PRIMARY KEY
	Name	Tên tàu	varchar	25	NOT NULL
	Seat_Sleeper	Số ghế giường nằm	int	4	NOT NULL
	Seat_First_Class_AC	Số ghế hạng nhất	int	4	NOT NULL
	Seat_Second_Class_AC	Số ghế hạng hai	int	4	NOT NULL
	Seat_Third_Class_AC	Số ghế hạng ba	int	4	NOT NULL
	Wifi	Có wifi	char	6	NOT NULL
	Food	Có thức ăn	char	6	NOT NULL
	Run_On_*	Chạy vào (thứ 2-Chủ nhật)	char	6	NOT NULL
Tickets	Ticket_No	Mã vé	int	x	NOT NULL, AUTO INCREMENT,

					PRIMARY KEY
	Coach	Số mã toa	varchar	2	NOT NULL
	Seat_No	Số ghế	char	2	NOT NULL
	Train_No	Mã tàu	int	x	NOT NULL
	Date_of_Journey	Ngày khởi hành	date	x	NOT NULL
	Username	Tên người dùng đặt vé	varchar	15	NOT NULL
Passenger	Passenger_Id	Mã hành khách	int	x	NOT NULL, AUTO INCREMENT, PRIMARY KEY
	First_Name	Họ	varchar	20	NOT NULL
	Last_Name	Tên	varchar	20	NOT NULL
	Gender	Giới tính	char	10	NOT NULL
	Phone_No	Số điện thoại liên lạc	char	10	DEFAULT NULL
	Ticket_No	Mã vé	int	x	NOT NULL
	Age	Tuổi	int	x	NOT NULL
	Class	Hạng ghế	varchar	20	NOT NULL
Station	Station_Code	Mã ga	char	5	NOT NULL, DEFAULT ‘’, PRIMARY KEY
	Station_Name	Tên ga	varchar	25	NOT NULL
Stoppage	Train_No	Mã tàu	int	x	NOT NULL, DEFAULT

					0, PRIMARY KEY
	Station_Code	Mã ga	char	5	NOT NULL, DEFAULT '', PRIMARY KEY
	Arrival_Time	Thời gian đến	time	x	NOT NULL
	Departure_Time	Thời gian đi	time	x	NOT NULL
Contact	Username	Tên khách hàng	varchar	15	NOT NULL, FOREIGN KEY, PRIMARY KEY
	Phone_No	Số điện thoại	char	10	PRIMARY KEY

VI. CÁC RÀNG BUỘC DỮ LIỆU

- Khóa ngoại ở bảng Contact

+ **Username tham chiếu tới Username trong bảng Account:**

```
ALTER TABLE Contact
ADD CONSTRAINT Contact_fk_Username
FOREIGN KEY (Username) REFERENCES Account (Username)
ON DELETE CASCADE;
```

- Khóa ngoại ở bảng Ticket

+ **Username tham chiếu tới Username trong bảng Account:**

```
ALTER TABLE Ticket
ADD CONSTRAINT Ticket_fk_Username
FOREIGN KEY (Username) REFERENCES Account (Username)
ON DELETE CASCADE;
```

+ **Train_No tham chiếu tới Train_No trong bảng Train:**

```
ADD CONSTRAINT Ticket_fk_Train_No
FOREIGN KEY (Train_No) REFERENCES Train (Train_No)
ON UPDATE CASCADE;
```

- Khóa ngoại ở bảng Passenger

+ **Ticket_No tham chiếu tới Ticket_No trong bảng Ticket:**

```
ALTER TABLE Passenger
ADD CONSTRAINT Passenger_fk_Ticket_No
FOREIGN KEY (Ticket_No) REFERENCES Ticket (Ticket_No)
ON DELETE CASCADE;
```

- Khóa ngoại ở bảng Stoppage

+ **Train_No tham chiếu tới Train_No trong bảng Train:**

```
ALTER TABLE Stoppage
ADD CONSTRAINT Stoppage_fk_Train_No
FOREIGN KEY (Train_No) REFERENCES Train (Train_No)
ON DELETE CASCADE ON UPDATE CASCADE,
```

+ **Station_Code tham chiếu tới Station_Code trong bảng Station:**

```
ADD CONSTRAINT Stoppage_fk_Station_Code
FOREIGN KEY (Station_Code) REFERENCES Station (Station_Code)
ON DELETE CASCADE ON UPDATE CASCADE;
```

VII. CÁC CÀI ĐẶT VẬT LÍ

1. Triggers

– File triggers.sql

+ **Kiểm tra ngày khởi hành có trước ngày hiện tại không:**

```
DELIMITER $$
CREATE TRIGGER before_date_of_journey
BEFORE UPDATE ON ticket
FOR EACH ROW
BEGIN
  IF NEW.date_of_journey <= CURDATE() THEN
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'Ngay khoi hanh phai lon hon ngay hien tai';
  END IF;
END$$
DELIMITER ;
```

+ **Kiểm tra địa chỉ email có hợp lệ không:**

```
DELIMITER $$
CREATE TRIGGER check_valid_email
BEFORE UPDATE ON account
FOR EACH ROW
BEGIN
  IF NEW.Email_Id NOT LIKE '%@gmail.com' THEN
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'Email khong hop le';
  END IF;
END$$
DELIMITER ;
```

+ **Kiểm tra số điện thoại có hợp lệ không:**

```
DELIMITER $$
CREATE TRIGGER check_valid_phone
BEFORE UPDATE ON contact
FOR EACH ROW
BEGIN
```

```
    IF NEW.Phone_No NOT LIKE '0%' AND LENGTH(NEW.Phone_No) !=
     10 THEN
       SIGNAL SQLSTATE '45000'
       SET MESSAGE_TEXT = 'So dien thoai khong hop le';
    END IF;
END$$
DELIMITER ;
```

+ **Kiểm tra thời gian đến có khớp với thời gian rời đi không**

```
DELIMITER $$
CREATE TRIGGER check_arrival_departure
BEFORE UPDATE ON stoppage
FOR EACH ROW
BEGIN
   IF NEW.Arrival_Time >= NEW.Departure_Time THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Thoi gian khong hop le';
   END IF;
END$$
DELIMITER ;
```

+ **Thêm chỗ mới trong bảng Train mỗi khi có vé bị hủy**

```
DELIMITER $$
CREATE TRIGGER cancel
BEFORE DELETE ON ticket
FOR EACH ROW
BEGIN
   SET @train = OLD.Train_No;
   SET @ticket = OLD.Ticket_No;
   SET @class = (SELECT class FROM passenger WHERE Ticket_No =
    @ticket);

   IF @class = 'Sleeper' THEN
      UPDATE train
      SET Seat_Sleeper = Seat_Sleeper + 1
      WHERE Train_No = @train;
   ELSEIF @class = 'First Class AC' THEN
```

```
    UPDATE train
    SET Seat_First_Class_AC = Seat_First_Class_AC + 1
    WHERE Train_No = @train;
  ELSEIF @class = 'Second Class AC' THEN
    UPDATE train
    SET Seat_Second_Class_AC = Seat_Second_Class_AC + 1
    WHERE Train_No = @train;
  ELSEIF @class = 'Third Class AC' THEN
    UPDATE train
    SET Seat_Third_Class_AC = Seat_Third_Class_AC + 1
    WHERE Train_No = @train;
  END IF;
END $$
DELIMITER ;
```

+ **Giảm số chỗ mỗi khi có có vé được được đặt:**

```
DELIMITER $$
CREATE TRIGGER add_ticket
AFTER INSERT ON ticket
FOR EACH ROW
BEGIN
  SET @train = NEW.Train_No;
  SET @ticket = NEW.Ticket_No;
  SET @class = (SELECT class FROM passenger WHERE Ticket_No =
   @ticket);

  IF @class = 'Sleeper' THEN
    UPDATE train
    SET Seat_Sleeper = Seat_Sleeper - 1
    WHERE Train_No = @train;
  ELSEIF @class = 'First Class AC' THEN
    UPDATE train
    SET Seat_First_Class_AC = Seat_First_Class_AC - 1
    WHERE Train_No = @train;
  ELSEIF @class = 'Second Class AC' THEN
```

```
        UPDATE train
        SET Seat_Second_Class_AC = Seat_Second_Class_AC - 1
        WHERE Train_No = @train;
    ELSEIF @class = 'Third Class AC' THEN
        UPDATE train
        SET Seat_Third_Class_AC = Seat_Third_Class_AC - 1
        WHERE Train_No = @train;
    END IF;
END $$
DELIMITER ;
```

2. Transactions

– File transaction.sql

+ **1. Đặt vé mới và cập nhật hành khách**

```
START TRANSACTION;

-- Thêm một vé mới vào bảng Ticket
INSERT INTO Ticket (Coach, Seat_No, Date_of_Journey, Username,
    Train_No)
VALUES ('D1', '40', '2024-12-17', 'nguoidung5', 202);

-- Giả sử Ticket_No vừa thêm là 8 (lấy bằng LAST_INSERT_ID())
-- Thêm một hành khách tương ứng vào bảng Passenger
INSERT INTO Passenger (First_Name, Last_Name, Gender, Phone_No,
    Ticket_No, Age, Class)
VALUES ('Nguyễn Văn', 'Hùng', 'Nam', '0912345678', LAST_INSERT_ID(),
    25, 'Second Class AC');

-- Kiểm tra lỗi và quyết định commit hay rollback
ROLLBACK;
-- Kết thúc giao dịch
```

+ **2. Xóa một vé và các dữ liệu liên quan**

```
START TRANSACTION;

-- Xóa hành khách liên quan trong bảng Passenger
DELETE FROM Passenger WHERE Ticket_No = 19;
```

```
-- Xóa vé khỏi bảng Ticket
DELETE FROM Ticket WHERE Ticket_No = 19;
-- Kiểm tra lỗi và quyết định commit hay rollback
ROLLBACK;

-- Kết thúc giao dịch
```

+ **3. Thay đổi thông tin tàu và các điểm dừng liên quan**

```
START TRANSACTION;

-- Cập nhật thông tin tàu trong bảng Train
UPDATE Train
SET Name = 'Tàu SE1 Mới'
WHERE Train_No = 202;

-- Cập nhật các điểm dừng trong bảng Stoppage
UPDATE Stoppage
SET Arrival_Time = '12:00:00', Departure_Time = '12:15:00'
WHERE Train_No = 202 AND Station_Code = 'ST01';

-- Kiểm tra lỗi và quyết định commit hay rollback
ROLLBACK;

-- Kết thúc giao dịch
```

+ **4. Cập nhật lịch trình hành trình**

```
START TRANSACTION;

-- Cập nhật thời gian đến tại ga tàu trong bảng Stoppage
UPDATE Stoppage
SET Arrival_Time = '14:30:00'
WHERE Train_No = 202 AND Station_Code = 'ST02';

-- Kiểm tra lỗi và quyết định commit hay rollback
ROLLBACK;
```

```
-- Kết thúc giao dịch
```

+ 5. Thêm thông tin tài khoản người dùng và liên hệ

```
START TRANSACTION;

-- Thêm một tài khoản mới vào bảng Account
INSERT INTO Account (Username, Password, Email_Id, Address)
VALUES ('nguoidung8', 'password123', 'nguoidung6@example.com', 'Q. Cau
    Giay');

-- Thêm thông tin liên hệ vào bảng Contact
INSERT INTO Contact (Username, Phone_No)
VALUES ('nguoidung8', '0987654321');

-- Kiểm tra lỗi và quyết định commit hay rollback
ROLLBACK;

-- Kết thúc giao dịch
```

3. Procedure

```
-- File: Procedure.sql
```

+ Thêm tài khoản mới

```
DELIMITER $$
CREATE PROCEDURE AddAccount(
   IN p_Username VARCHAR(15),
   IN p_Password VARCHAR(20),
   IN p_Email_Id VARCHAR(35),
   IN p_Address VARCHAR(50)
)
BEGIN
   INSERT INTO Account (Username, Password, Email_Id, Address)
   VALUES (p_Username, p_Password, p_Email_Id, p_Address);
END$$
DELIMITER ;
```

+ Cập nhật thông tin tàu

```
DELIMITER $$
CREATE PROCEDURE UpdateTrain(
   IN p_Train_No INT,
   IN p_Name VARCHAR(25),
   IN p_Seat_Sleeper INT,
   IN p_Seat_First_Class_AC INT,
   IN p_Seat_Second_Class_AC INT,
   IN p_Seat_Third_Class_AC INT,
   IN p_Wifi CHAR(6),
   IN p_Food CHAR(6),
   IN p_Run_On_Sunday CHAR(6),
   IN p_Run_On_Monday CHAR(6),
   IN p_Run_On_Tuesday CHAR(6),
   IN p_Run_On_Wednesday CHAR(6),
   IN p_Run_On_Thursday CHAR(6),
   IN p_Run_On_Friday CHAR(6),
   IN p_Run_On_Saturday CHAR(6)
)
BEGIN
   UPDATE Train
   SET Name = p_Name,
      Seat_Sleeper = p_Seat_Sleeper,
      Seat_First_Class_AC = p_Seat_First_Class_AC,
      Seat_Second_Class_AC = p_Seat_Second_Class_AC,
      Seat_Third_Class_AC = p_Seat_Third_Class_AC,
      Wifi = p_Wifi,
      Food = p_Food,
      Run_On_Sunday = p_Run_On_Sunday,
      Run_On_Monday = p_Run_On_Monday,
      Run_On_Tuesday = p_Run_On_Tuesday,
      Run_On_Wednesday = p_Run_On_Wednesday,
      Run_On_Thursday = p_Run_On_Thursday,
      Run_On_Friday = p_Run_On_Friday,
      Run_On_Saturday = p_Run_On_Saturday
   WHERE Train_No = p_Train_No;
```

```
END$$
DELIMITER ;
```

-- Đặt vé mới

```
DELIMITER $$
CREATE PROCEDURE BookTicket(
    IN p_Coach VARCHAR(2),
    IN p_Seat_No CHAR(2),
    IN p_Date_of_Journey DATE,
    IN p_Username VARCHAR(15),
    IN p_Train_No INT,
    IN p_First_Name VARCHAR(20),
    IN p_Last_Name VARCHAR(20),
    IN p_Gender CHAR(10),
    IN p_Phone_No CHAR(10),
    IN p_Age INT,
    IN p_Class VARCHAR(20)
)
BEGIN
    START TRANSACTION;
    -- Thêm vé mới
    INSERT INTO Ticket (Coach, Seat_No, Date_of_Journey, Username,
     Train_No)
    VALUES (p_Coach, p_Seat_No, p_Date_of_Journey, p_Username,
     p_Train_No);

    -- Lấy Ticket_No vừa thêm
    SET @Ticket_No = LAST_INSERT_ID();

    -- Thêm hành khách tương ứng
    INSERT INTO Passenger (First_Name, Last_Name, Gender, Phone_No,
     Ticket_No, Age, Class)
    VALUES (p_First_Name, p_Last_Name, p_Gender, p_Phone_No,
     @Ticket_No, p_Age, p_Class);

    COMMIT;
END$$
```

```
DELIMITER ;

-- Hủy vé
DELIMITER $$
CREATE PROCEDURE CancelTicket(
   IN p_Ticket_No INT
)
BEGIN
   START TRANSACTION;
   -- Xóa hành khách liên quan
   DELETE FROM Passenger WHERE Ticket_No = p_Ticket_No;

   -- Xóa vé
   DELETE FROM Ticket WHERE Ticket_No = p_Ticket_No;

   COMMIT;
END$$
DELIMITER ;
```

4. Queries

– File queries.sql

+ **Tên các tàu dừng tại Ga Hà Nội**

```
SELECT t.Name
FROM train t
INNER JOIN stoppage sp ON t.Train_No = sp.Train_No
INNER JOIN station st ON sp.Station_Code = st.Station_Code
WHERE st.Station_Name = "Ga Hà Nội" ;
```

	Name
▶	SE1

+ **Thông tin về các hành khách trên tàu SE1**

```
SELECT p.*
FROM passenger p
```

```
INNER JOIN ticket tk ON p.Ticket_No = tk.Ticket_No
INNER JOIN train t ON tk.Train_No = t.Train_No
WHERE t.Name = "SE1" ;
```

Passenger_Id	First_Name	Last_Name	Gender	Phone_No	Ticket_No	Age	Class
1	Nguyễn Hồng	Phúc	Nam	0901234567	1	19	Second Class AC

+ **Thông tin các tài khoản và mã vé tương ứng tài khoản đó đặt, kể cả khi không đặt vé nào**

```
SELECT a.*, tk.Ticket_No
FROM account a
LEFT OUTER JOIN ticket tk ON a.Username = tk.Username ;
```

Username	Password	Email_Id	Address	Ticket_No
nguoidung1	matkhau1	nguoidung1@gmail.com	Số 1, P. Hoàn Kiếm	1
nguoidung2	matkhau2	nguoidung2@gmail.com	Số 2, TP. Vinh	2
nguoidung3	matkhau3	nguoidung3@gmail.com	Số 3, Q. Hải Châu	3
nguoidung4	matkhau4	nguoidung4@gmail.com	Số 4, Q. Tân Bình	4
nguoidung5	matkhau5	nguoidung5@gmail.com	Số 5, P. Đông Hà	5
nguoidung6	matkhau6	nguoidung6@gmail.com	Số 6, TP. Huế	6
nguoidung7	matkhau7	nguoidung7@gmail.com	Số 7, Q. Bình Thạnh	7

+ **Số hiệu, tên tàu và số ga (trong danh sách 6 ga) mà nó đến, kể cả khi không có**

```
SELECT t.Train_No, t.Name, COUNT(sp.Station_Code) AS
Number_of_Destinations
FROM train t
LEFT OUTER JOIN stoppage sp ON t.Train_No = sp.Train_No
GROUP BY t.Train_No ;
```

	Train_No	Name	Number_of_Destinations
▶	201	SE1	2
	202	SE2	1
	203	TN1	1
	204	TN2	1
	205	SN1	1
	206	SN2	0
	207	SE3	0

+ **Họ tên các hành khách có chuyến tàu vào ngày 13-12-2024**

```
SELECT CONCAT(p.First_Name, " ", p.Last_Name) AS Full_Name
FROM passenger p
WHERE p.Ticket_No IN
  (SELECT t.Ticket_No FROM ticket t WHERE t.Date_of_Journey =
"2024-12-13") ;
```

	Full_Name
▶	Nguyễn Đình Quyền

+ **Số điện thoại của tài khoản đặt vé đi tàu TN2**

```
SELECT c.Phone_No FROM contact c WHERE c.Username IN
  (SELECT t.Username FROM ticket t WHERE t.Train_No IN
        (SELECT tr.Train_No FROM train tr WHERE tr.Name = "TN2"));
```

	Phone_No
▶	0904567890

+ **Mã vé của các vé "có vinh dự" vào các tàu có sức chứa > 300**

```
SELECT Ticket_No, Total_Seat
FROM (SELECT tk.Ticket_No, (t.Seat_Sleeper + t.Seat_First_Class_AC
```

```
                + t.Seat_Second_Class_AC + t.Seat_Third_Class_AC) AS
Total_Seat
        FROM ticket tk
    INNER JOIN train t ON tk.Train_No = t.Train_No) AS SeatTicket
WHERE Total_Seat > 300 ;
```

	Ticket_No	Total_Seat
▶	1	340
	2	400
	4	305
	7	460

+ **Mã vẽ của các vé có chủ sở hữu trong họ và tên có chữ 't'**

```
SELECT Ticket_No, Full_Name
FROM (SELECT tk.Ticket_No, CONCAT(p.First_Name, " ", p.Last_Name) AS
Full_Name
        FROM ticket tk
    INNER JOIN passenger p ON tk.Ticket_No = p.Ticket_No) AS Ticket_Owner
WHERE Full_Name LIKE "%t%" ;
```

	Ticket_No	Full_Name
▶	2	Trần Doãn Thắng
	6	Vũ Thị F

+ **Tổng số hành khách trên các chuyến tàu có tích hợp Wifi**

```
SELECT COUNT(p.Passenger_Id) AS Number_of_POWCT
FROM passenger p
INNER JOIN ticket tk ON p.Ticket_No = tk.Ticket_No
INNER JOIN train t ON tk.Train_No = t.Train_No
WHERE t.Wifi = "Có" ;
```

	Number_of_POWCT
▶	4

+ **Số hành khách đến các ga**

```
SELECT st.Station_Name, COUNT(p.Passenger_Id) AS Number_of_P
FROM station st
INNER JOIN stoppage sp ON st.Station_Code = sp.Station_Code
INNER JOIN train t ON sp.Train_No = t.Train_No
INNER JOIN ticket tk ON t.Train_No = tk.Train_No
INNER JOIN passenger p ON tk.Ticket_No = p.Ticket_No
GROUP BY st.Station_Code ;
```

	Station_Name	Number_of_P
▶	Ga Hà Nội	1
	Ga Thanh Hóa	1
	Ga Vinh	1
	Ga Đồng Hới	1
	Ga Huế	1
	Ga Đà Nẵng	1

VIII. KẾT LUẬN VÀ TIỀM NĂNG PHÁT TRIỂN

Hệ thống của chúng tôi đã thành công trong việc cung cấp thông tin đa dạng về các chuyến tàu, hỗ trợ tìm kiếm tàu giữa hai ga, cũng như đặt và hủy vé. Nhờ vậy, hệ thống có tiềm năng ứng dụng trong hoạt động đặt vé tàu chính thức. Tuy nhiên, để hoàn thiện hơn nữa, chúng tôi dự định bổ sung thêm nhiều tính năng hữu ích, chẳng hạn như dịch vụ đặt suất ăn trên tàu. Bên cạnh đó, việc tích hợp các cổng thanh toán là ưu tiên hàng đầu nhằm đảm bảo an toàn tuyệt đối cho các giao dịch.