• **flightScheduler()** sẽ lên lịch cho số lượng chuyến bay cần thiết, ví dụ: 10, 20 chuyến bay ngẫu nhiên.
• **addCustomerToFlight()** sẽ thêm khách hàng vào danh sách khách hàng hiện tại của chuyến bay hiện tại.
• **calculateFlightTime()** tính thời gian chuyến bay hoặc thời gian máy bay đã di chuyển.
• **fetchArrivalTime()** tính thời gian ngày giờ khi máy bay chạm đất.
• **deleteFlight()** sẽ xóa chuyến bay được liệt kê trong lịch trình chuyến bay. (Chỉ có quản trị viên mới có thể thực hiện)
• **calculateDistance()** tính khoảng cách giữa các sân bay của điểm xuất phát và điểm đến (được chọn ngẫu nhiên từ mảng 2D) tính bằng dặm và kilômét.
• **distanceMeasurementInstructions()** hiển thị một số hướng dẫn thông thường, cách thời gian chuyến bay được tính toán, khi nào nên đến sân bay, v.v...
• **displayFlightSchedule()** hiển thị lịch trình chuyến bay đã tạo. Lịch trình chuyến bay thay đổi mỗi khi chương trình chạy.
• **createNewFlightsAndTime()** tạo mới chuyến bay với các thuộc tính cần thiết của nó và sau đó gọi phương thức **getNearestHourQuarter()** để làm tròn thời gian đến phút gần nhất của giờ.

Flight

- flightSchedule:String
- flightNumber:String
- fromWhichCity:String
- gate:String
- toWhichCity:String
- distanceInMiles:double
- distanceInKm:double
- flightTime:String
- numOfSeatsInTheFlight:int
- listOfRegisteredCustomersInTheFlight:List
- customerIndex:int
- flightList:List
- nextFlightDay:int

+ Flight() + Flight(String, String, int, String[][], String[], String)
+ flightScheduler():void
+ addNewCustomerToFlight(Customer):void
+ addTicketsToExistingCustomer(Customer, int):void
+ isCustomerAlreadyRegistered(List, Customer):boolean
+ calculateFlightTime(double):String
+ fetchArrivalTime():String
+ deleteFlight():void
+ calculateDistance(double, double, double, double):String[]
+ radianToDegree():double
+ degreeToRadian(double):double
+ displayFlightSchedule():void
+ toString(int):String
+ createNewFlightsAndTime():String
+ getNearestHourQuarter(LocalDateTime):LocalDateTime
+ settersAndGetters():

<<abstract>>
FlightDistance

+ toString(int):String
+ calculateDistance(double, double, double, double):String[]
+ distanceMeasurementInstructions():void

has flight

• **isUniqueData()** trả về true nếu tên người dùng hoặc email đã tồn tại trong danh sách .
• **addFlightToCustomerList()** sẽ kết hợp chuyến bay đó với khách hàng hiện tại.

Customer

- userID:String
- name:String
- email:String
- password:String
- address:String
- phone:String
- age:int
- flightsRegisteredByUser:List
- numOfTIcketsBookedByUser:List
- customerCollection:List

+ Customer()
+ Customer(String, String, String, String, String, int)
+ addNewCustomer():void
+ isUniqueData(String, String):boolean
+ editUserInfo(String):void
+ deleteUser(String):void
+ searchUser(String):void
+ toString(int):String
+ addNewFlightToCustomerList(Flight):void
+ addExistingFlightToCustomerList(int, int):void
+ displayCustomersData():void
+ displayHeader():void
+ randomIDDisplay():String
+ displayArtWork():void
+ settersAndGetters():

1..*
has registered
0..*

FlightReservation

- c1:Customer
- flight:Flight
- flightIndexInFlightList:int

+ FlightReservation()
+ FlightReservation(FlightReservation)
+ bookFlight(String, int, String):void
+ cancelFlight(String, String):void
+ addNumberOfTicketsToExistingBookedFlight(Customer, int):void
+ addNumberOfTicketsForNewFlight(Customer, int):void
+ addNumberOfTicketsToUser(List, Flight, Customer, int):void
+ isFlightAlreadyAddedToCustomerList(List,Flight):boolean
+ flightStatus():boolean
+ equals():boolean
+ toString(int, Flight, Customer):String
+ flightIndex(List, Flight):int
+ displayFlightsRegisteredByOneUser(String):void
+ toString(int, Customer):String
+ displayHeaderForUsers(Flight, List):void
+ displayRegisteredUsersForAllFlights():void
+ displayRegisteredUsersForASpecificFlight(String):void
+ displayArtWork(int):void
+ settersAndGetters(): <respectiveReturnType>

customer has an ID

RandomGenerator

- randomNum:String
- destinations:String[][]

- randomIDGen():String
- randomDestinations():String[][]
- randomNumOfSeats():int
- randomFlightNumGen(int, int):Stringg

User

- adminUserNameAndPassword:String[][]
- customersCollection:List

+ displayMainMenu():void
+ mannualInstructions():void
+ welcomScreen(int):void
+ printArtWork(int):void
+ getCustomersCollection():List

can be

is user registered

RolesAndPermissions

+ isPrivilligedUserOrNot(String, String):int
+ isPassengerRegistered(String, String):String

<<Interface>>
DisplayClass

+ displayRegisteredUsersForAllFlights():void
+ displayRegisteredUsersForASpecificFlight():void
+ displayHeaderForUsers():void
+ displayFlightsRegisteredByOneUser():void
+ displayArtWork():void

• **bookFlight()** đặt chỗ cho người dùng trong chuyến bay đó.
• **cancelFlight()** hủy đăng ký của người dùng trong chuyến bay đó.
• **flightStatus()** sẽ kiểm tra xem chuyến bay có đúng theo lịch trình hay đã bị hủy bỏ không.
• **isFlightAlreadyAddedToCustomer()** sẽ kiểm tra xem chuyến bay hiện tại đã được liên kết với người dùng hiện tại hay chưa.
• **displayFlightsRegisteredByOneUser()** sẽ hiển thị tất cả các chuyến bay được đăng ký bởi một người dùng.
• **displayRegisteredUsersForAllFlights()** sẽ hiển thị tất cả hành khách đã đăng ký cho tất cả các chuyến bay.
• **displayRegisteredUsersForASpecificFlight()** sẽ hiển thị tất cả hành khách đã đăng ký cho một chuyến bay cụ thể.