

**Table of Contents**

[**1 Introduction**](#_ohosgjq03xi9) **4**

[1.1 Objective](#_1zslly2nwffn) 4

[1.2 Scope](#_6gvi3452sxpb) 4

[1.3 Glossary](#_339q4f7jgxhf) 4

[1.4 References](#_me9jaahys4ny) 4

[**2 Overall Description**](#_j20eyyfl3p4d) **4**

[2.1 General Overview](#_ehc3jmjnx1q8) 5

[2.2 Assumptions/Constraints/Risks](#_hqt2bd9rv51) 5

[2.2.1 Assumptions](#_x5chdff8p8gk) 5

[2.2.2 Constraints](#_2c69kgf7e7pj) 5

[2.2.3 Risks](#_gw1eg7cjmpcl) 6

[**3 System Architecture and Architecture Design**](#_k4cclt8syyr3) **6**

[3.1 Architectural Patterns](#_bq2rh57qs6o0) 6

[3.2 Interaction Diagrams](#_51grokr4b4gd) 6

[3.3 Analysis Class Diagrams](#_jga58oeb17lh) 6

[3.4 Unified Analysis Class Diagram](#_r75tn0wtwfrm) 6

[3.5 Security Software Architecture](#_an7hmve3l74o) 6

[**4 Detailed Design**](#_xrxmexi5d6yx) **6**

[4.1 User Interface Design](#_z3jt2m82rc2i) 6

[4.1.1 Screen Configuration Standardization](#_g52s412ecgjl) 6

[4.1.2 Screen Transition Diagrams](#_ezoqtyapcg5) 7

[4.1.3 Screen Specifications](#_zaxld5wf5a9s) 7

[4.2 Data Modeling](#_ol7jyd1uwhl) 12

[4.2.1 Conceptual Data Modeling](#_gc4td36byuwp) 12

[4.2.2 Database Design](#_qh0laqt5oaw) 13

[4.2.2.1 Database Management System](#_rbd0tf4f3ge3) 13

[4.2.2.2 Database Diagram](#_cj7ug2xj8g0n) 13

[4.2.2.3 Database Detail Design](#_mmgvajvtjdas) 14

[4.3 Non-Database Management System Files](#_4jeozpygpplb) 18

[4.4 Class Design](#_x5hokskv7j28) 18

[4.4.1 Class Diagrams](#_13xtloz6m9we) 18

[4.4.2 Class Design](#_gdq5f3m2lgiv) 18

[4.4.2.1 Class “BookingController”](#_m82va8lr96x) 18

[4.4.2.2 Class “FlightController”](#_h6wzi7hyikke) 19

[4.4.2.3 Class “LoginController”](#_sp2i6zvdfby0) 20

[4.4.2.4 Class “RegistrationController"](#_6mfkq9yby3ej) 20

**List of Figures**

# 

# **1** **Introduction**

Travelwala is the pioneering travel and lifestyle app in Southeast Asia. Its purpose is to assist people in discovering and purchasing a wide variety of travel products, local services, and financial services. Travelwala's diverse product portfolio includes airline tickets, buses, trains, car rentals, and airport transfers, as well as the largest hotel and accommodation inventory in Southeast Asia. Apart from that, we also offer a variety of local attractions, activities, wellness spas, and beauty spas to help people fulfill several of their luxury dreams. As such, whatever your lifestyle preferences are, you're only one click away!

## ***1.1*** ***Objective***

This SDD is written for the purpose of giving the audience a clear view about the design of the software. The document’s intended audience is stakeholder and software designer/developer.

## ***1.2*** ***Scope***

<*In this subsection:*

*(1)*  *Identify the software product(s) to be produced by name*

*(2)*  *Explain what the software product(s) will, and, if necessary, will not do*

*(3)*  *Describe the application of the software being specified, including relevant benefits, objectives, and goals*

*(4)*  *Be consistent with similar statements in higher-level specifications if they exist*

*This should be an executive-level summary. Do not enumerate the whole requirements list here*

*Note that this will be similar to what was written in the SRS.*

>

## ***1.3*** ***Glossary***

*<Listing and explaining the terms appearing in the software’s profession and this document. Any assumption of the reader’s prior knowledge or experience on the subject is ill advised>*

## ***1.4*** ***References***

| [1] | Centers for Medicare & Medicaid Services, "System Design Document Template," [Online]. Available: https://www.cms.gov/Research-Statistics-Data-and-Systems/CMS-Information-Technology/XLC/Downloads/SystemDesignDocument.docx. |
| --- | --- |

# 

# **2** **Overall Description**

## ***2.1*** ***General Overview***

Concerning the system, we have some features that make the apps resemble e-commerce websites or software: We have an interface for interacting with the user; user requests are made by clicking on the interface, and the request is then processed by the system controller; we have a database (remote) to store any type of data; any data-related request or change is queried in the database, and the change in the database is then reflected in the UI; we have a database (remote) to store any type of data; any data-related request or change is queried in the database, and (user interface). As you can see, the system is comprised of three major components: the user interface (UI), the controller, and the data model. We decided to make this software a desktop application. The three-layer architecture was chosen as our design approach. The design architecture aids in the separation of different components and the organization of the codebase.

Here is general use-case diagram to help you understand the core of our design:

*Fig 1. General Use Case Diagram*

## ***2.2*** ***Assumptions/Constraints/Risks***

### **2.2.1** **Assumptions**

Users who use the software should have a good connection to the Internet. Also, our software is a desktop application, so the user also must have a laptop/desktop with an OS (we recommended 64 Bit Microsoft Windows 8 or later; macOS 10.13 or later; or any Linux distribution that supports running applications) to run the apps. About the system requirement, we would say 2 GB RAM minimum, 8 GB RAM recommended; for storage 2.5 GB and another 1 GB for caches minimum, solid-state drive with at least 5 GB of free space recommended; require latest version of JRE; 1024×768 minimum screen resolution, 1920×1080 is a recommended screen resolution.

### **2.2.2** **Constraints**

*<Describe any global limitations or constraints that have a significant impact on the design of the system’s hardware, software and/or communications, and describe the associated impact. Such constraints may be imposed by any of the following (the list is not exhaustive):*

· *Hardware or software environment*

· *End-user environment*

· *Availability or volatility of resources*

· *Standards compliance*

· *Interoperability requirements*

· *Interface/protocol requirements*

· *Licensing requirements*

· *Data repository and distribution requirements*

· *Security requirements (or other such regulations)*

· *Memory or other capacity limitations*

· *Performance requirements*

· *Network communications*

· *Verification and validation requirements (testing)*

· *Other means of addressing quality goals*

· *Other requirements described in the Requirements Document*

*>*

### **2.2.3** **Risks**

*<Describe any risks associated with the system design and proposed mitigation strategies.>*

# 

# **3** **System Architecture and Architecture Design**

Architecture Design steps:

1. Find out software components -> use cases

2. Find out Interaction between use cases

3. Find out Relationship between use cases

4. Draw UML Diagram includes: interaction diagram and analysis class diagram

## ***3.1*** ***Architectural Patterns***

In our project, we use a 3-tier architectural pattern.

## ***3.2*** ***Interaction Diagrams***

## ***3.3*** ***Analysis Class Diagrams***

## ***3.4*** ***Unified Analysis Class Diagram***

## ***3.5*** ***Security Software Architecture***

# 

# **4** **Detailed Design**

## ***4.1*** ***User Interface Design***

### **4.1.1** **Screen Configuration Standardization**

***Display***

Number of colors supported: 16,777,216 colors Resolution: 1366x768 𝑝𝑖𝑥𝑒𝑙𝑠

***Screen***

Location of standard buttons: At the bottom (vertically) and in the middle (horizontally) of the frame

Location of the messages: Starting from the top vertically and in the middle horizontally of the frame down to the bottom.

Display of the screen title: The title is located at the top of the frame in the middle.

Consistency in expression of alpha numeric numbers: comma for separator of thousand while strings only consist of characters, digits, commas, dots, spaces, underscores, and hyphen symbol.

***Control***

Size of the text: medium size (mostly 24px). Font: Segoe UI. Color: #000000

Input check process: Should check if it is empty or not. Next, check if the input is in the correct format or not

Sequence of moving the focus: There will be no stack frames. Each screen will be separated. However, the manual is considered a popup message, as the main screen cannot be operated while the manual screen is shown. After the opening screen, the app will start with a splash screen, and then the first screen (home screen) will appear.

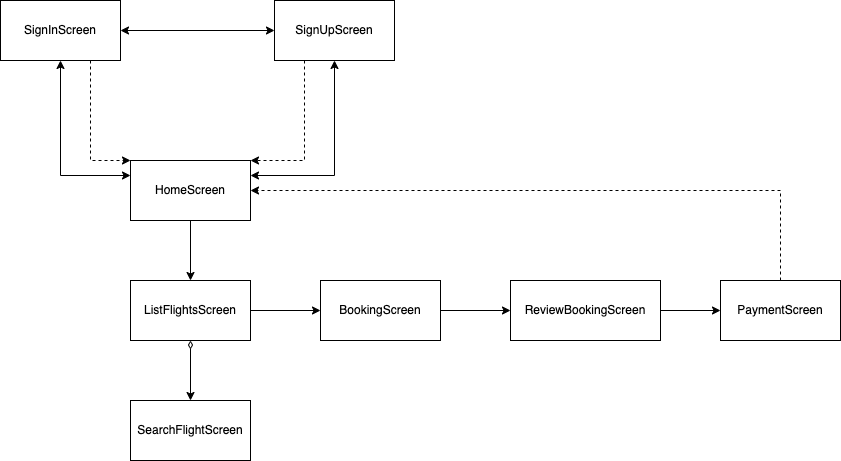
***Direct input from the keyboard***

There will be no shortcuts. There are back buttons to move back to the previous screen. Also, there is the close button “X” located at the title bar to the right to close the screen.

***Error***

A message will be given to notify the users what is the problem.

### **4.1.2** **Screen Transition Diagrams**



### **4.1.3** **Screen Specifications**

| Travelwala Web-app | | Date of creation | Approved by | Reviewed by | Person in charge |
| --- | --- | --- | --- | --- | --- |
| Screen specification | Home Screen | 08/02/2023 | Tô Xuân Hùng | Tô Xuân Hùng | Nguyễn Minh Quân |
|  | | Control | Operation | Function | |
| Area for displaying the Navigation bar | Initial | Enable user to Navigate between sites | |
| Area for displaying flight options boxes | Initial | Display list of options for users to choose to configure their flights | |
| Location box | Click | Display departure and arrival locations for user to choose for their flights | |
| Departure Datepicker | Click | Choose departure date for the flight | |
| Arrival Check box | Click | Choose the Check box to toggle the Arrival Date box. | |
| Arrival Datepicker | Click | Choose arrival date for the flight | |
| Number of passengers box | Click | Choose number of passengers for the booking | |
| Seat class box | Click | Choose the class for the flight seats | |
| Search flight button | Click | Submit all information from the forms above to search for appropriate flights | |
| Footer Area | Click | Choose icons to navigate to Wala Corp social media sites | |

| Travelwala Web-app | | Date of creation | Approved by | Reviewed by | Person in charge |
| --- | --- | --- | --- | --- | --- |
| Screen specification | Sign In Screen | 08/02/2023 | Tô Xuân Hùng | Tô Xuân Hùng | Nguyễn Minh Quân |
|  | | Control | Operation | Function | |
| Area for displaying the Navigation bar | Initial | Enable user to Navigate between sites | |
| Book Now button | Click | Click to navigate to Home page | |
| Email and Password text field | Type | Input login credentials | |
| Remember me button | Click | Click to toggle Remember login information for further logins | |
| Register here button | Click | Click to navigate to Sign up page | |
|  | | Login with Google button | Click | Click to navigate to Google OAuth Sign in page | |
|  | | Login with Facebook button | Click | Click to navigate to Facebook OAuth Sign in page | |

| Travelwala Web-app | | Date of creation | Approved by | Reviewed by | Person in charge |
| --- | --- | --- | --- | --- | --- |
| Screen specification | Sign Up Screen | 08/02/2023 | Tô Xuân Hùng | Tô Xuân Hùng | Nguyễn Minh Quân |
|  | | Control | Operation | Function | |
| Area for displaying the Navigation bar | Initial | Enable user to Navigate between sites | |
| Book Now button | Click | Click to navigate to Home page | |
| Area of text fields to take user credentials | Type | Input user account credentials | |
| Sign Up button | Click | Click to submit Sign Up form and navigate to Home page | |
| Login button | Click | Click to navigate to Sign in page | |
|  | | Login with Google button | Click | Click to navigate to Google OAuth Sign in page | |

| Travelwala Web-app | | Date of creation | Approved by | Reviewed by | Person in charge |
| --- | --- | --- | --- | --- | --- |
| Screen specification | Booking Screen | 08/02/2023 | Tô Xuân Hùng | Tô Xuân Hùng | Nguyễn Minh Quân |
|  | | Control | Operation | Function | |
| Area for displaying the Navigation bar | Initial | Enable user to Navigate between sites | |
| Text field area to input user contact details | Type | Take contact information from user | |
| Text field area to input travelers credentials | Type | Take credentials from users. Number of these text field areas is mapped from the number of passengers taken from the Home page. | |
| Continue to Review Booking button | Click | Click to navigate to Review Booking page | |

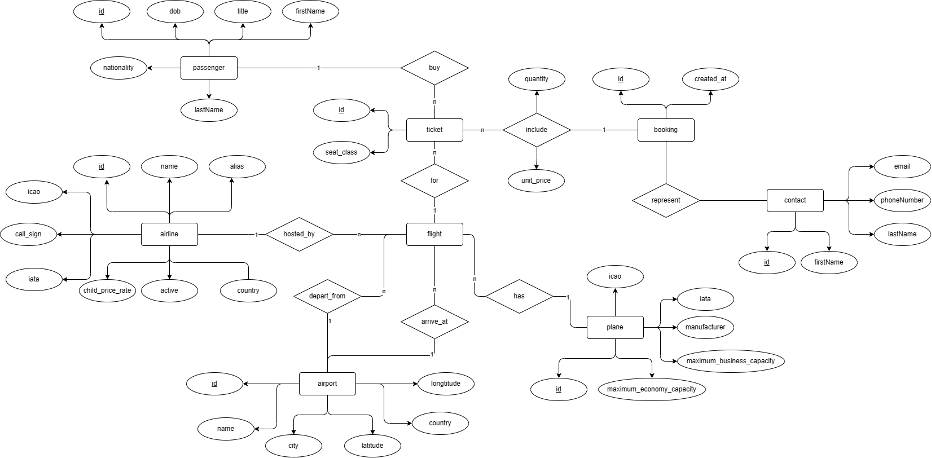
| Travelwala Web-app | | Date of creation | Approved by | Reviewed by | Person in charge |
| --- | --- | --- | --- | --- | --- |
| Screen specification | Review Booking Screen | 08/02/2023 | Tô Xuân Hùng | Tô Xuân Hùng | Nguyễn Minh Quân |
|  | | Control | Operation | Function | |
| Area for displaying the Navigation bar | Initial | Enable user to Navigate between sites | |
| Contact details Box | Initial | Display Contact information, taken from Booking page | |
| Travelers credentials Box | Initial | Display travelers credentials, taken from Booking page | |
| Edit detail buttons | Click | Toggle Contact details Box and Travelers credentials Box to text field to edit user information | |
| Price detail box | Initial | Display total price of the whole booking (all the users flight ticket price, hotels and discount coupon if there is | |
|  | | Continue to payment button | Click | Click to navigate to Payment page | |

| Travelwala Web-app | | Date of creation | Approved by | Reviewed by | Person in charge |
| --- | --- | --- | --- | --- | --- |
| Screen specification | List Flight screen | 14/01/2023 | Tô Xuân Hùng | Tô Xuân Hùng | Pham Duc Minh |
|  | | Control | Operation | Function | |
| Area for displaying navigation bar | Initial | navigate to different screens | |
| Area for displaying flight search info | Initial | display chosen search options for flights | |
| Area for displaying departure/return flights | Initial | Display the suitable flights based on search | |
| Flight cell | Click | choose flight for departure/return | |
| Area for displaying chosen flights | Initial | display flights chosen by the user | |
|  | | Continue booking button | Click | transition to the booking screen | |
|  | | Change search button | Click | activate search popup | |
|  | | Footer area | Click | choose icons to navigate to Wala Corp social media sites | |

| Travelwala Web-app | | Date of creation | Approved by | Reviewed by | Person in charge |
| --- | --- | --- | --- | --- | --- |
| Screen specification | Search Popup screen | 14/01/2023 | Tô Xuân Hùng | Tô Xuân Hùng | Pham Duc Minh |
|  | | Control | Operation | Function | |
| Area for displaying the Navigation bar | Initial | Enable user to Navigate between sites | |
| Area for displaying flight options boxes | Initial | Display list of options for users to choose to configure their flights | |
| Location box | Initial | Display departure and arrival locations for user to choose for their flights | |
| Departure Datepicker | Click | Choose departure date for the flight | |
| Arrival Check box | Click | Choose the Check box to toggle the Arrival Date box. | |
|  | | Arrival Datepicker | Click | Choose arrival date for the flight | |
|  | | Number of passengers box | Click | Choose number of passengers for the booking | |
|  | | Seat class box | Click | Choose the class for the flight seats | |
|  | | Search flight button | Click | Submit all information from the forms above to search for appropriate flights | |

## ***4.2*** ***Data Modeling***

### **4.2.1** **Conceptual Data Modeling**



### **4.2.2** **Database Design**

#### ***4.2.2.1*** ***Database Management System***

MongoDB is a NoSQL database management system that has higher speed and higher

performance than relational database management system (RDBMS). There is no data

constraint in MongoDB, hence we do not need to spend time on checking data constraints as we do in RDBMS.

Most SQL databases require scaling up vertically (migrate to a larger, more expensive

server) when exceeding the capacity requirements of your current server. Conversely,

most NoSQL databases allow us to scale-out horizontally, meaning we can add cheaper,

commodity servers whenever we need to.

Queries in NoSQL databases can be faster than SQL databases. Data in SQL databases is

typically normalized, so queries for a single object or entity require you to join data from

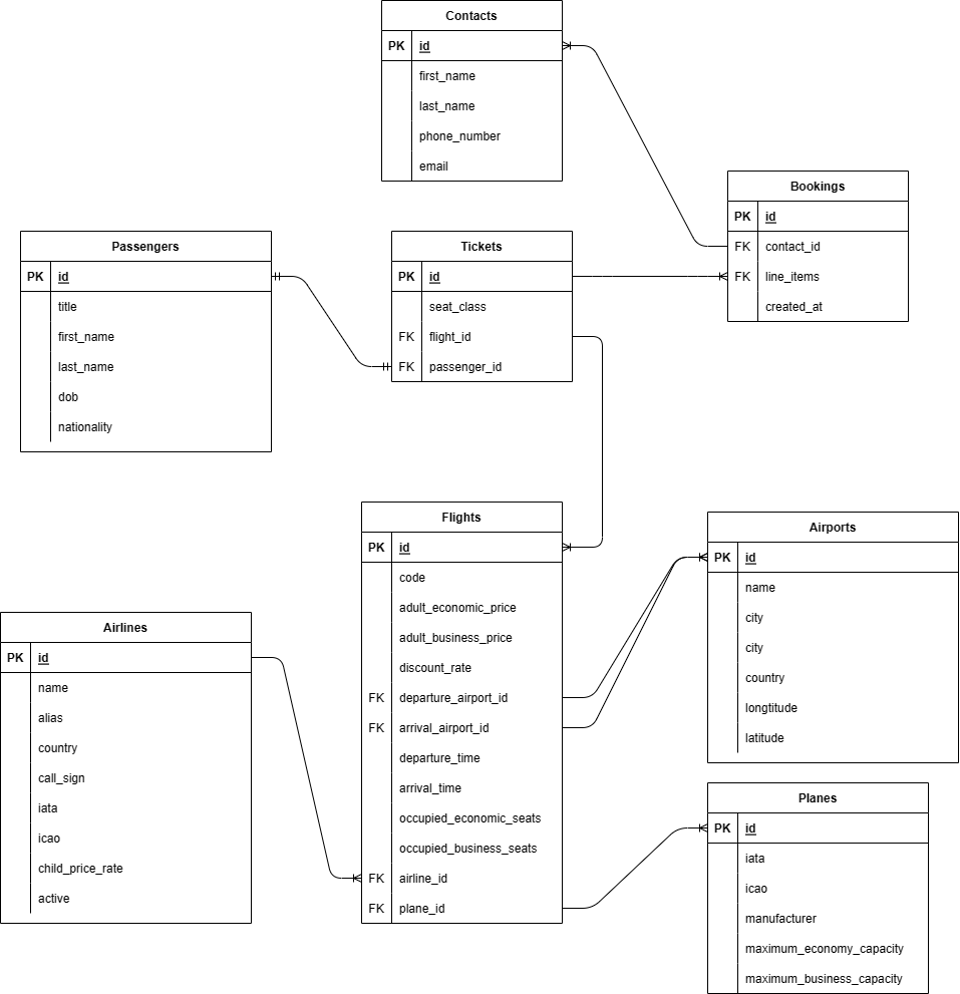
multiple tables. As tables grow, the joins can become expensive. However, data

in NoSQL databases is typically stored in a way that is optimized for queries. The rule of

thumb when using MongoDB is data that is accessed together should be stored together.

Queries typically do not require joins, so the queries are very fast.

#### ***4.2.2.2*** ***Database Diagram***



#### ***4.2.2.3*** ***Database Detail Design***

· **Table Design**

- **flights**

| *#* | *PK* | *Column name* | *Data type* | *Description* |
| --- | --- | --- | --- | --- |
| 1 | x | id | ObjectId | ID of the flight |
| 2 |  | adult\_economic\_price | Double | Price for an adult economic seat |
| 3 |  | adult\_business\_price | Double | Price for an adult business seat |
| 4 |  | discount\_rate | Double | Discount rate |
| 5 |  | departure\_airport | Object | Departure airport |
| 6 |  | arrival\_airport | Object | Arrival airport |
| 7 |  | departure\_time | Timestamp | Departure time |
| 8 |  | arrival\_time | Timestamp | Arrival time |
| 9 |  | occupied\_economic\_seats | Double | Currently occupied economic seats |
| 10 |  | occupied\_business\_seats | Double | Currently occupied business seats |
| 11 |  | airline | Object | Airline that takes care of the flight |
| 12 |  | plane | Object | Plane that is used for the flight |

- **airports**

| *#* | *PK* | *Column name* | *Data type* | *Description* |
| --- | --- | --- | --- | --- |
| 1 | x | id | ObjectId | ID of the airport |
| 2 |  | name | String | Name of the airport |
| 3 |  | city | String | City in which the airport is located |
| 4 |  | country | String | Country in which the airport is located |
| 5 |  | longitude | Double | Longitude of the airport |
| 6 |  | latitude | Double | Latitude of the airport |

- **airlines**

| *#* | *PK* | *Column name* | *Data type* | *Description* |
| --- | --- | --- | --- | --- |
| 1 | x | id | ObjectId | ID of the airline |
| 2 |  | alias | String | Short name of the airline |
| 3 |  | country | String | Country that the airline belongs to |
| 4 |  | call\_sign | String | International call sign of the airline |
| 5 |  | iata | String | International IATA code of the airline |
| 6 |  | icao | String | International ICAO code of the airline |
| 7 |  | child\_price\_rate | Double | Price rate for a child according to adult price |
| 8 |  | active | Boolean | Whether the airline is active |

- **planes**

| *#* | *PK* | *Column name* | *Data type* | *Description* |
| --- | --- | --- | --- | --- |
| 1 | x | id | ObjectId | ID of the plane |
| 2 |  | iata | String | International IATA code of the plane |
| 3 |  | icao | String | International ICAO code of the plane |
| 4 |  | manufacturer | String | Manufacturer of the plane |
| 5 |  | maximum\_economy\_capacity | Integer | Maximum seats for economy class |
| 6 |  | maximum\_business\_capacity | Integer | Maximum seats for business class |

- **passengers**

| *#* | *PK* | *Column name* | *Data type* | *Description* |
| --- | --- | --- | --- | --- |
| 1 | x | id | ObjectId | ID of the passenger |
| 2 |  | first\_name | String | First name of the passenger |
| 3 |  | last\_name | String | Last name of the passenger |
| 4 |  | dob | Timestamp | Date of birth of the passenger |
| 5 |  | nationality | String | Nationality of the passenger |
| 6 |  | title | String | Title of the passenger: Mr, Ms |

- **contacts**

| *#* | *PK* | *Column name* | *Data type* | *Description* |
| --- | --- | --- | --- | --- |
| 1 | x | id | ObjectId | ID of the contact |
| 2 |  | first\_name | String | First name of the contact |
| 3 |  | last\_name | String | Last name of the contact |
| 4 |  | phone\_number | String | Phone number of the contact |
| 5 |  | email | String | Email of the contact |

- **tickets**

| *#* | *PK* | *Column name* | *Data type* | *Description* |
| --- | --- | --- | --- | --- |
| 1 | x | id | ObjectId | ID of the ticket |
| 2 |  | seat\_class | String | Seat class of the order (economic, business) |
| 3 |  | flight\_id | ObjectId | ID of the flight for the ticket |
| 4 |  | passenger\_id | ObjectId | ID of the passenger that books this ticket |

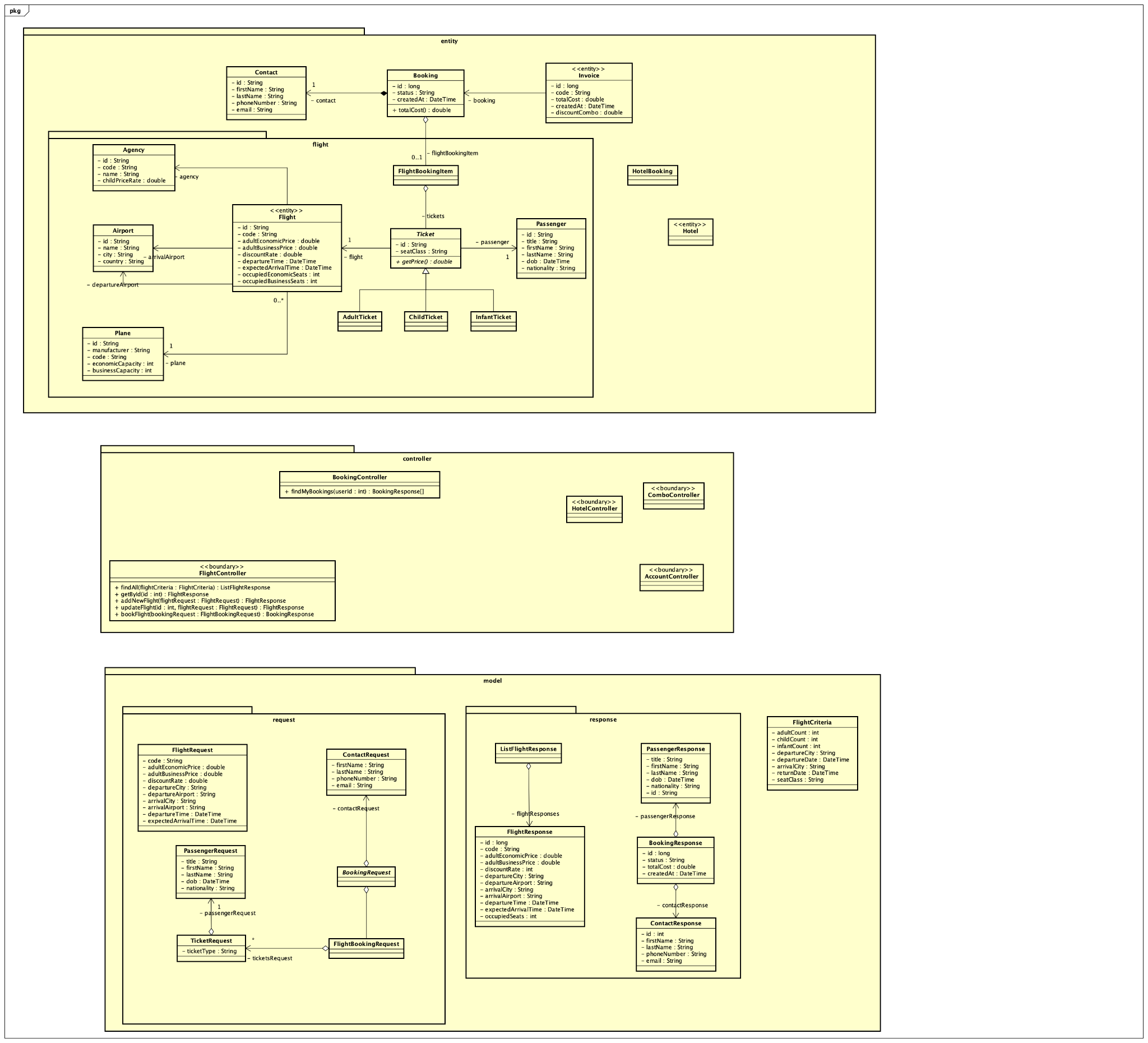
- **bookings**

| *#* | *PK* | *Column name* | *Data type* | *Description* |
| --- | --- | --- | --- | --- |
| 1 | x | id | ObjectId | ID of the booking |
| 2 |  | contact\_id | ObjectId | The representative of the booking |
| 3 |  | line\_items | Array | Array of tickets for the booking |
| 4 |  | created\_at | Timestamp | Time when the booking is processed |

## ***4.3*** ***Non-Database Management System Files***

## ***4.4*** ***Class Design***

### **4.4.1** **Class Diagrams**



### 

### **4.4.2** **Class Design**

#### **4.4.2.1 Class “BookingController”**

**Attribute**

createBooking: an booking interface

**Operation**

| *#* | *Name* | *Return type* | *Description (purpose)* |
| --- | --- | --- | --- |
| 1 | book | ResponseEnitity<Booking> | Return the booking that user already booked |

*Parameter*:

* bookingRequest: BookingRequest, a booking request from user

*Exception*:

**Method**

None

**State**

None

#### **4.4.2.2 Class “FlightController”**

**Attribute**

flightService: FlightService class to handle logic related to flights

**Operation**

| *#* | *Name* | *Return type* | *Description (purpose)* |
| --- | --- | --- | --- |
| 1 | findAll | ResponseEnitity<SearchFlightResponse> | Return all the flights that user find |
| 2 | findById | ResponseEnitity<FlightResponse> | Return flight that user find |
| 3 | createNewFlight | ResponseEnitity<FlightResponse> | Return flight that user created |
| 4 | updateFlight | ResponseEnitity<FlightResponse> | Return flight that user updated |

*Parameter*:

* id: String, an identifier of a flight
* flightCriteria: FlightCriteria, criterias of an flight
* flightRequest: FlightRequest, request of flight

*Exception*:

None

**Method**

None

**State**

None

#### **4.4.2.3 Class “LoginController”**

**Attribute**

None

**Operation**

| *#* | *Name* | *Return type* | *Description* |
| --- | --- | --- | --- |
| 1 | userRegistration | ResponseEntity | Receive registration requests from users, create and store users’ information to the database. |
| 2 | confirm | ResponseEntity | Triggered when users confirm their account through email. |

*Parameter*:

* userData: users’ information provided by users to register an account.
* token: an unique identifier for a registration’s email.

*Exception*:

* RuntimeException if old access token is missing or invalidated

**Method**

None

**State**

None

#### **4.4.2.4 Class “RegistrationController"**

**Attribute**

None

**Operation**

| *#* | *Name* | *Return type* | *Description* |
| --- | --- | --- | --- |
| 1 | refreshToken | void | Get refresh token |

*Parameter*:

* request: HTTP request from user
* response: HTTP response, returned back to user

*Exception*:

* UserAlreadyExistException if users provide existing identifier information (username, email, telephone).
* IllegalStateException if there exists errors while confirming register email.

**Method**

None

**State**

None