Software Engineering

Group 9

Design of the Flight Booking Subsystem for a Holiday Booking Website

Catherine Solari - 1688142 - ces642
Daniel Hill - 1700739 - djh639
Rachel Waterman - 1715647 - rjw649
Thomas Brereton - 1708846 - tlb646

Introduction

The aim of this project is to engineer a sensible and effective architecture for a sub-system of the BookMeHolidays website. This online system allows both corporate users and the general public to search for and book flights, hotels and holiday packages as well as providing a review sharing function, discussion forums and a recommendations system. This project focuses on the subsystem which books flights once a user has selected the flight they wish to book. The scope of this sub-system is discussed below along with the assumptions made about the other subsystems with which the flight booking system will need to interact.

Scope of the System

The flight booking subsystem is activated when a user has selected a flight which they wish to book using the search subsystem. In selecting a particular flight, the information already specified about the booking and thus available to the booking subsystem includes:

- Flight departure destination
- Flight arrival destination
- Flight operator
- Flight departure date and time
- Ticket class (First, business, economy etc)
- Number of passengers
- Base price of the ticket including taxes

Whilst a user may be able to search for flights whilst not registered with or signed into the system, they cannot book and pay for a flight without being logged into the website. The booking system must therefore ensure that a user is signed in before they can confirm the booking with the flight operator. The booking system should have access to the user database which can provide the following information:

- User account information – supplied by the user database
    - Corporate users – Account identifier, contact number, contact email address, bank account details
    - Non-corporate user – Account identifier, contact email address

The subsystem is required to collect the additional information required to make the booking and transfer the necessary information to the relevant parties, whilst ensuring that the information is stored safely and securely. The system is also required to offer possible add-ons such as in-flight meals, insurance and seat selection to the customer and ensure their selections are booked correctly with the required parties. In order to fulfil the required functionality, the information that must be collected by the system includes:

- Customer passport information – Full name, nationality, D.O.B, passport number and date of expiration
- Seat selection
- In-flight meals – including dietary requirements
- Additional luggage – including oversize items
- Travel insurance

The booking system is required to provide relevant parties and accompanying subsystems with the information they require to validate the booking, this will allow any security checks to take place as well as enabling the customer's specifications to be officially confirmed:

- Customer passport information must often be shared with security services from the country of departure and sometimes of arrival to confirm a passenger's flight status
- The flight operator requires the customer information and preferences in terms of meals, luggage and seat selection in order to enter the booking into their systems and confirm the booking. The flight operator will also push any changes to flights to the booking system, which must be able to determine which users must be informed of the changes
- The insurer providing the travel insurance requires the customer information and some of the travel details in order to confirm the travel insurance
- An email subsystem requires the booking reference and customer contact email in order to send a confirmation of the booking to the customer once it has been confirmed with the flight operator.
- The hotel booking system must run concurrently if a package holiday is being booked
- The recommender system requires the flight destination and travel dates as well as the number of passengers in order to recommend features such as hotels and car hire which might be of interest to the user

It is customary to give a customer a cancellation period in which they can cancel their booking at no additional cost, this is normally to the order of an hour or so, this must be taken into account by the booking system with the booking being editable and cancellable for an hour after the user has submitted it. Concurrency and scalability must also be taken into consideration; there are likely to be periods in which the website will experience heavy user traffic, for example leading up to school holidays and the system must be scalable to allow for a large number of users to access the system at the same time without servers crashing and booking errors occurring. The system must also be designed to reduce the chance of duplicate bookings, which again is more likely to occur during busy periods where multiple users may be trying to book the same flight, causing a risk of the flight becoming overbooked. It is unlikely that this website will be the only one advertising any specific flight and the system must account for the possibility of duplicate bookings across other websites and travel agents, the flight operator will likely be the only one who knows the number of booking that have already been made for a particular flight and the booking system of BookMeHolidays must take this into account. In terms of the security of the system, it must be ensured that any data is stored according to the rules of the region in which the database and servers are located. Data protection and privacy laws of various countries should be considered and where possible, data should be stored in the home region of a user, flight operator, insurance provider etc to reduce the chance of data usage by unwanted parties. This could mean that different parts of the flight booking are stored in different regions and as such, the system must provide a method of keeping track of all parts of the booking, likely by creating a booking reference.

Assumptions for related sub-systems

- Users of the system must have an account with the website in order to be able to book flights. Given that there are two types of customer, namely corporate users who pay for use of the system and individuals who can use it for free, it can be assumed that there are two types of accounts. A corporate account will have an account identifier, contact information and payment information in order to collect the monthly

subscription fee. An individual account need only have an account identifier and a contact email address as no subscription fee is required. In terms of the booking system, it does not matter whether the user is a corporate user or not. It is assumed that this user data is stored in a secure database that is accessible by the booking system.

- Payment is handled by a separate sub-system, it is assumed that the booking system will determine the method of payment required and transfer the user to the correct sub-system to process payment.
- It is assumed that all other parts of the system are scalable with respect to users, additional products and the addition of new security checks. It is also assumed that the rest of the system has sufficient back-up provision for business continuity in the case of main server or database errors. This includes the database containing the user account information specified previously and additional databases concerning hotel and package booking details. Servers specific to the flight booking sub-system are considered to be part of the scope of this project, the actual databases used are not directly part of the sub-system however they are not a system in themselves and act only as a storage hub.
- The main system will contain a separate email subsystem which will process requests from the booking system to contact the relevant user with their booking confirmation and any updates on their booking. These updates will be pushed to the booking system by the flight operator for any flight that is changed and the booking system must determine which users must be informed and pass these details to the email system (the email system does not have access to the bookings databases to increase security and reduce coupling of the system)
- It is assumed that the seat allocation process is performed by a different subsystem, a call to which will allow the user to pick from currently available seats.

Summary of System Scope

The booking system will cover all aspects of booking a flight from the point at which the user has selected a specific flight and travel class and pressed book. The booking system will not cover payment, seat allocation or the actual sending of emails to the user however it will send requests to the relevant sub systems to perform these actions. The booking system will be responsible for maintaining the record of which users have booked which flights by accessing the booking databases as well as actually booking the user a place on the plane with the relevant airline. The booking system will also provide functionality to select additional options for the flight, including seat reservations, extra luggage and meals and convey these selections to the relevant flight operator. The system should offer the user travel insurance and send a request to the insurance sub-system if insurance is to be added. The flight booking system will also contact the recommender system with flight information in order to allow the recommender system to recommend other products which may be of interest to the user.

Requirements Analysis

Functional Requirements

1. User accounts

a) A user may be able to browse for and select flights before having to sign in to the system however they must be both registered and signed in to be able to book a flight.

- The system must check that a user is signed in to their account before they can purchase a flight and enter passenger information
- The system should transfer the user to the user account management sub system if they are not signed in or do not have an account.
- The system must only proceed with a booking once a user has signed in so that fraudulent bookings cannot occur

2. Passenger and Flight Security

a) The system must ensure all passengers have a legal flight status

- The system will send passenger passport information to the flight security services of the country of departure to ensure that the passenger is allowed to leave the country.

b) The system could alert the user to visa requirements in their destination country

- The system could provide the user with instructions on how to obtain any visa required
- The system could inform the user of eligibility requirements for a visa before their booking is confirmed

3. Confirmation and Updates

a) The system must enable confirmation emails to be sent to users after a successful booking

- The system will confirm with the flight operator that the booking has been made
- The system will send a request with the booking information and the user to contact to the email subsystem

b) The system must enable the user to be notified if any changes are made to their flight

- The flight operator's system will push any flight changes to the booking system
- The system must determine any users with a booking for that flight
- The system must send a request to the email subsystem to contact the relevant users with the update from the flight operator

4. Flight add-ons

a) The system should give the user the opportunity to order an inflight meal on flights where this is possible

- The system should check with the flight operator if inflight meals are available
- The system could allow the user to specify dietary requirements

b) The system should allow the user to add additional baggage

- The system must check with the flight operator if additional large items can be added to the booking
- The system must determine the additional cost of adding the extra baggage
- The system could allow the user to select from different types of baggage - bikes, sports gear etc.

c) The system should give the user the option to pre-select their seat

- The system should check with the flight operator which seats on the plane are currently unassigned

- The system should display all eligible seats for the user to pick from
- The system must then relay these selections to the flight operator as part of the booking

5. Insurance

a) The system could offer the user the functionality to add travel insurance to their booking

- The system could send a request to the insurance subsystem to get quotes for insurance for the flight being booked
- If the user selects to buy insurance, the insurance subsystem could run concurrently and the user's selections should be added to the booking

6. Concurrency

a) The system must prevent multiple users from booking the same seat simultaneously and causing overbooking of the flight or duplicate bookings

- The system will request a provisional booking on the flight selected by the user with the flight operator until the booking has been completed

b) The system must allow the user to cancel the flight at no charge for 1 hour after the booking is confirmed

- The system will store the time the booking is made as part of the booking. Once the cancellation hour has passed, the booking databases will be updated with the confirmed booking.

7. Payment

a) The system must determine the total to be paid by the user and record the price breakdown as part of the booking

- The total to pay will include the base price of the ticket (including taxes) - to be paid to the flight operator
- The cost of any add ons provided by the flight operator will be included - to be paid to the flight operator
- The cost of any insurance will be included- to be paid to the insurance company

b) The system must determine the preferred payment method of the user

- Corporate customers will already have payment details registered as part of their user account however an option for them to pay via a different means will be offered
- For non-corporate users, payment by credit or debit card should be offered.
- Payment using paypal or bank transfer could also be available

c) The system must transfer the user to the relevant payment system

- The system should send a request to the relevant payment sub-system with a total to be received by the user as well as a price breakdown so that the payment system can handle the payments to the third parties such as the flight operator

8. Links to Hotel Booking and Recommendations Systems

a) The system should run concurrently with the hotel booking system if a package deal is being purchased

- All input user data that is required by both the flight and hotel booking systems will be input into the flight booking system and shared with the hotel booking system. The hotel system will then collect any additional information it requires.
- Both the flight and hotel bookings will be confirmed with the travel operators concurrently with the package not being confirmed with the user until both aspects of the package have been confirmed.

Non-Functional Requirements

1. Security

a) The system must ensure that all data collected is stored securely to prevent access by unwanted parties

- The booking databases will be located regionally - (Americas, Europe, Asia etc)
- Data should be stored according to the nationality of the user making the booking and the home region of the flight operator
- Data pertaining to a particular booking may be spread across several databases and as such must be retrievable via a common booking reference.

b) The system should send only the minimum required information to any other sub-system or third party

- Passenger information for security checks should only be sent to the country of departure if legally required.
- The email system should only receive the user's email address and booking specification, no other account details such as bank account details should be shared.

2. Reliability

a) The system should have safeguards in place to minimise effects of component malfunction

- Each regional booking database should have a duplicate back up system which is synced regularly
- Databases and back-up databases should be automatically synced after every 100 bookings or every hour in case of system malfunction

3. Scalability

a) The system should be constructed such that it can be scaled for an increased number of users

- The system should utilise multiple identical servers, each of which can provide all the functionalities that the booking system provides
- A load balancer should control server traffic and prevent server overload by splitting users between all available servers proportional to their capacity.
- The system should be designed such that additional databases and servers can be easily added in order to expand the system.

4. Process Efficiency

a) The system should minimise user wait times during the booking process where possible

- The load balancer could split users between all available servers proportionally to their capacity and/or speed.
- The system should contact the flight operator's systems to make a provisional booking as soon as a logged in user tries to book a selected flight
- The system could request add-ons from the flight operator/other third party as soon as they are requested, if feasible, rather than compiling the booking and requesting all in one go
- The system should query availability of add ons such as meals and specific seat selection as soon as the user wishes to book to reduce wait times when selecting add ons

5. Maintainability

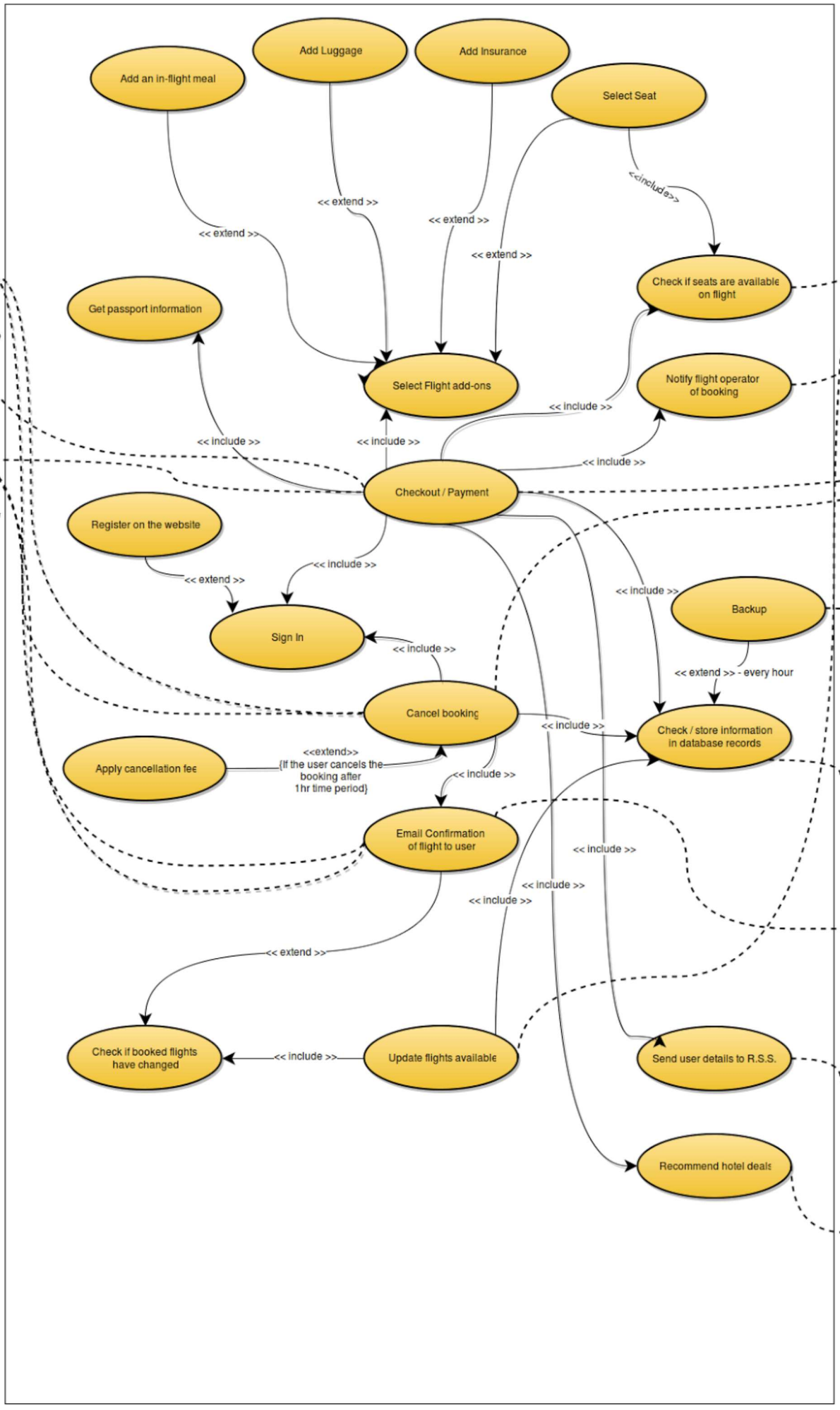
a) The system should be able to run autonomously with minimal human oversight

- Each server should be able to offer full functionality individually to reduce impact of servers being taken offline during any required system maintenance
- Each server should have a separate link to the flight operators and other third parties to reduce bottlenecks in the system and improve reliability

UML Design of the System

Use-case diagram

The Use Case Diagram for the flight booking system is presented on the next page. The actors that have been identified are shown outside the boundary of the system, which is displayed by the rectangle that surrounds the diagram. Within the boundaries are the use cases along with the relationships that the use cases have both with the main actors and each other are also displayed. Additionally, three use cases have been selected for further analysis with pre and post-conditions, actors, flow of events and a possible scenario being identified.


Add Luggage
Add Insurance
Add an in-flight meal
Select Seat
<<include>>
<< extend >>
<< extend >>
<< extend >>
<< extend >>
Flight Operator
Corporate Account Holder
Get passport information
Check if seats are available on flight
Select Flight add-ons
Notify flight operator of booking
<< include >>
<< include >>
<< include >>
Individual Account Holder
Checkout / Payment
Payment Subsystem
Register on the website
<< include >>
<< extend >>
Sign In
<< include >>
<< include >>
Backup
Backup Booking Database
<< extend >> - every hour
Cancel booking
<< include >>
Check / store information in database records
Apply cancellation fee
<<extend>>
{If the user cancels the booking after 1hr time period}
<< include >>
Regional Booking Database
Email Confirmation of flight to user
<< include >>
<< include >>
<< include >>
Email Subsystem
<< extend >>
Check if booked flights have changed
<< include >>
Update flights available
Send user details to R.S.S.
Regional Security Server(R.S.S.)
Recommend hotel deals
Hotel Booking Subsystem

Documented use cases:

Three use cases have been selected for further documenting. These are:
1.Book a British Airways flight for an individual account holder.
2.Book a flight for a corporate account holder with 5 passengers.
3.Cancel a booking after a two hour time period for an individual account holder.

These are detailed below, showing the relevant actors, preconditions, event flow and post-conditions.

Use Case Description One:
Book a British Airways flight for an individual account holder.

Precondition:
The customer is an individual account holder.
The customer hasn't registered.
The customer has selected a flight with British Airways from London to Barcelona.
The customer has not entered payment details or contact number.

Flow Of Events:

1. The use case starts when the user has selected a flight and is at the checkout/payment page.
2. The system first checks if seats are available.
    a. If no seats are available the customer is asked to select a different flight.
    b. If seats are available the customer continues checking out.
3. The system notifies the flight operator of the booking to temporarily reserve a seat.
4. The system prompts the customer to select an in flight meal.
    a. If the user selects a meal the selection is sent to the Flight Operator
    b. If the user does not select a mean nothing is sent
5. The system gives the customer an option to add additional luggage.
    a. If the user selects to add luggage the selection is sent to the Flight Operator.
    b. If the user does not select any additional luggage nothing is sent.
6. The system prompts the customer to add insurance.
    a. If the user selects to add insurance the selection is sent to the Flight Operator.
    b. If the user does not select any additional insurance nothing is sent.
7. The Hotel Booking subsystem advertises relevant hotel deals for the customer to select.
    a. If the user selects to add a hotel booking, the selection is sent to the Hotel Booking subsystem.
    b. If the user does not select a hotel booking, nothing is sent.
8. The system gives the customer an option to select a specific seat.
    a. If the selected seat is not available the customer is asked to select a different seat.
    b. If the specific seat is available the selection is sent to the Flight Operator.

9. The system asks the user to sign in or register
    a. If the customer has not registered they must register.
    b. If the customer has registered they must sign in.
10. The register page on the system asks the user if they would like to register as an individual or corporate account holder. The customer is an individual account holder therefore selects the option for individual account holders.
11. The account identifier and contact email address is saved in the databases
12. The system recognises they are registered and signed in.
13. The system asks for their passport information.
    a. The system sends the passport information to the R.S.S. to validate it in real time.
    b. The system send the passport information to the regional database.
    c. If the R.S.S doesn't validate the user is prompted to re-enter the details.
14. The system prompts the user to select payment type and enter their payment details.
    a. The system sends the payment details to the payment subsystem to validate it in real time.
    b. The system sends the payment details to the regional database.
    c. If the subsystem doesn't validate the user is prompted to re-enter the details.
15. The system asks the customer to confirm flight and process payment.
16. The system sends the payment details to the payment subsystem.
    a. If there are insufficient funds the user is asked to use a different card.
    b. If there are sufficient funds the booking is finalised
17. The system sends the final booking information to the regional booking database and the backup booking database.
18. The system send the confirmed flight details to the flight operator which permanently reserves a seat for the user.
19. The email subsystem sends a confirmation email to the customer.

.

Post Conditions:

1. The customer has been charged for the flight if a transaction was successful.
2. The system has told the flight operator to reserve seat(s) for the customer.
3. The customer has been cleared to fly by the regional security system database
4. The customer's details have been stored in the regional database.
5. The customer has received a confirmation email of the booking from the email subsystem.

Actors:

The primary actor involved in this use case is the Individual account holder who initiates it by selecting a flight and checking out. Also involved is the Flight Operator which keeps track if seats are available on each flight and if any flights are changed. The payment and passport information are checked and processed by the Payment Subsystem and Regional Security Server respectively. Moreover, during the checkout screen the Hotel Booking subsystem advertises deals for the customer to select. Lastly, the payment, passport, and account information are stored in the Regional and Backup Booking Database.

Scenario:

The customer named Alex has chosen to book a British airways flight from London to Barcelona with the system. He is required to register with the site as an individual account holder as he is not affiliated to a corporation. He is prompted to select the flight add ons which include adding an inflight meal, adding additional luggage, adding insurance. Also he is prompted to select a seat only if he wants a specific seat on the flight. He is required to

provide his passport details and credit card details to purchase the flight. The system checks in real time whether the passport and credit card information he has input is valid therefore the information are checked against the regional security server(R.S.S.) and payment subsystem. His credit card information is entered incorrectly and he is prompted to re-enter the details. He can see that the details he has entered are incorrect. He corrects the error and proceeds once again with the payment. This time the credit card details are valid and he is not wanted by the security services for any criminal activity therefore he can proceed with the booking. He pays with his visa debit card which has sufficient funds therefore the transaction is processed quickly without any errors. He is happy to see a confirmation email has been sent immediately so that he can view his booking with all the details required for him to book his transport to the airport.

Use Case Description Two:
Book a flight for a corporate account holder with 5 passengers.

Precondition:
The customer is a corporate account client that hasn't registered.
The customer has selected a flight with British Airways from London to Melbourne.
The customer has not entered payment details or contact number.

Flow Of Events:

1. The use case starts when the customer has selected a flight for 5 passengers and is at the checkout/payment page.
2. The system first checks if seats are available.
    a. If no seats are available the customer is asked to select a different flight.
    b. If seats are available the customer continues checking out.
3. The system notifies the flight operator of the booking to temporarily reserve 5 seats.
4. The system prompts the customer to select an in flight meal.
    a. If the user selects a meal, the selection is sent to the Flight Operator
    b. If the user does not select a meal, nothing is sent
5. The system gives the customer an option to add additional luggage.
    a. If the user selects to add luggage, the selection is sent to the Flight Operator.
    b. If the user does not select any additional luggage, nothing is sent.
6. The system prompts the customer to add insurance.
    a. If the user selects to add insurance, the selection is sent to the Flight Operator.
    b. If the user does not select any additional insurance, nothing is sent.
7. The Hotel Booking subsystem advertises relevant hotel deals for the customer to select.
    a. If the user selects to add a hotel booking, the selection is sent to the Hotel Booking subsystem.
    b. If the user does not select a hotel booking, nothing is sent.
8. The system gives the customer an option to select specific seats.
    a. If the selected seats are not available, the customer is asked to select different seats.
    b. If the specific seats are available the selection is sent to the Flight Operator.
9. The system asks the user to sign in or register
    a. If the customer has not registered they must register.
    b. If the customer has registered they must sign in.

10. The register page on the system asks the user if they would like to register as an individual or corporate account holder.
    a. The customer is an corporate account holder therefore selects the option for corporate account.
11. The system asks the corporate customer to register their contact number, contact email address, and payment details
    a. The payment subsystem validates the payment details.
    b. If the payment details are not valid the customer is asked to enter new details.
    c. If the payment details are valid, the customer continues.
12. The account identifier, contact email address, and payment details are saved in the databases
13. The system recognises they are registered and signed in.
14. The system asks for passport information of the passengers.
    a. The system sends the passport information to the R.S.S. to validate it in real time.
    b. The system sends the passport information to the regional database.
    c. If the R.S.S doesn't validate the user is prompted to re-enter the details.
15. The system prompts the user to select payment type.
    a. The system provides the option to pay with the saved or new payment details.
    b. If it is new payment details, the system sends the payment details to the payment subsystem to validate it in real time.
    c. If it is new payment details, the system sends the payment details to the regional database.
    d. If the subsystem doesn't validate the user is prompted to re-enter the details.
16. The system asks the customer to confirm flight and process payment.
17. The system sends the payment details to the payment subsystem.
    a. If there are insufficient funds the user is asked to use a different card.
    b. If there are sufficient funds the booking is finalised
18. The system sends the final booking information to the regional booking database and the backup booking database.
19. The system sends the confirmed flight details to the flight operator which permanently reserves the seats for the customer.
20. The email subsystem sends a confirmation email to the customer.

Post Conditions:

1. The customer has been charged for the flight if a transaction was successful.
2. The system has told the flight operator to reserve seat(s) for the customer.
3. The customer has been cleared to fly by the regional security system database
4. The customer's details have been stored in the regional database.
5. The customer has received a confirmation email of the booking from the email subsystem.

Actors:

The primary actor involved in this use case is the corporate account holder who initiates the process by selecting five seats on the flight and checking out. Furthermore, the Flight Operator that keeps track of the seats available on each flight is used and is also used if any of the flight specifications are changed. The payment and passport information are checked and processed by the Payment Subsystem and Regional Security Server, respectively. Moreover, during the checkout screen the Hotel Booking subsystem advertises deals for the

customer to select. Lastly, the payment, passport, and account information are stored in the Regional and Backup Booking Database.

Scenario:
The corporate account holder BT has chosen to book a British airways flight from London to Melbourne with the system. The company is sending five business executives in first class to attend a conference in Australia to promote a new IT service offered by the company. The manager of the finance department named Catherine is making the booking. Catherine is required to register with the site as a corporate account holder as BT is not currently registered on the site. She is prompted to select the flight add ons which will be added for all the passengers include adding an inflight meals, adding additional luggage, adding insurance for the group members. Moreover, she is prompted to select specific seats for the passengers. She is required to provide the passport details of all the passengers. She is also prompted to add the credit card details of BT which is linked to the expense account of the company. The transaction will be processed as a business purchase and the card she used is saved on the system's database and can be used for trips that BT make in the future on the system. The system checks in real time whether the passport and credit card information input are valid therefore the information is checked against the regional security server(R.S.S.) and payment subsystem. The credit card information is entered correctly therefore she proceeds with the payment. All of the passengers are clear to travel with the R.S.S therefore the booking is processed. The transaction is finally processed and a confirmation email is sent to BT's finance department email address which was used during the registration process. Catherine forwards the email to her supervisor so the company can keep a record of the purchase. Catherine also sends an email to all team members so they can plan their schedule around the trip.

Use Case Description Three:
Cancel a booking after a two hour time period for an individual account holder.

Precondition:
The customer is an individual account holder that has already registered.
The customer has selected a flight with RyanAir from Dublin to Madrid.
The customer has entered payment details and all the contact details.
A time period of two hours has passed after the booking.

Flow Of Events:
1. The use case starts when the customer has signed into the system and is shown the landing page with the details of the flights that user has booked.
2. The system displays options to cancel, change booking or make another booking.
    a. If the cancel button is selected the cancellation page is shown to the user.
    b. If the change booking button is selected the user is shown the change booking page.
    c. If the make another booking button is selected the user is shown a page to select a new flight.
3. The customer selects the cancel button and the cancellation page is displayed.
4. The system checks the time that has passed from the initial confirmed booking.
    a. If the time period is less than one hour no cancellation fee is applied.
    b. If the time period is greater than one hour a cancellation fee is applied.

5. The system displays any charges that will be applied due to the cancellation of the booking along with any reimbursements due to the customer.
6. The system displays an option for the user to either confirm the cancellation or make no flight cancellation.
    a. If the user selects confirm the cancellation, the cancellation is confirmed and the system along with all the databases and external actors are updated.
    b. If the user selects make no flight cancellation the customer is taken to the initial landing page with the details of the flight without modification or cancellation.
7. The customer is cancelling the flight therefore the confirm the cancellation button is selected.
8. The flight operator subsystem is updated and the pre-booked seat is made available.
    a. The conditions of the purchased flight are checked and the reimbursement due from the flight operator is calculated.
9. The payment subsystem is notified and the final balance is calculated from the reimbursement due and cancellation charges.
    a. If the time period is less than one hour no cancellation fee is applied.
    b. If the time period is greater than one hour a cancellation fee is applied.
    c. The reimbursement amount will depend on the conditions of the flight operator and the conditions that were confirmed upon purchasing the flight.
10. The database and Backup database and Regional Database are updated.
    a. The flight booking remains in the databases for a record however the booking is now cancelled so a marker is placed against the booking.
    b. The account of the customer remains in the database for future purchases.
11. All the information on the R.S.S. remains on the system for a period of two years.
12. The email subsystem sends an email to the customer's email account to notify them of the cancellation details along with all the financial information and updates.
13. The customer is notified by the system that the cancellation has been made.
    a. The email subsystem sends an email to the customer with a detailed breakdown of the charges, reimbursements and all other information related to the cancellation.
14. The customer is then redirected to the landing page of their account which is updated.
    a. If the customer has another flight booking it is displayed.
    b. If the customer does not have any other bookings the customer is informed they have no current bookings.
    c. The customer is prompted to make a new booking for future travel.

Post Conditions:

1. The customer has cancelled their flight.
2. The customer has been charged for the cancellation if 1 hour has occurred since making the initial booking.
3. The system has told the flight operator to de-allocate the reserved seat.
4. The system's database has been updated the customer record as flight cancelled
5. The customer has received a confirmation email of the cancellation from the email subsystem.

Actors:

The primary actor involved in this use case is the individual account holder who initiates the process by signing in and selecting cancel booking. Furthermore, the Flight Operator is notified when the cancellation is made so the seat(s) are deallocated. The payment

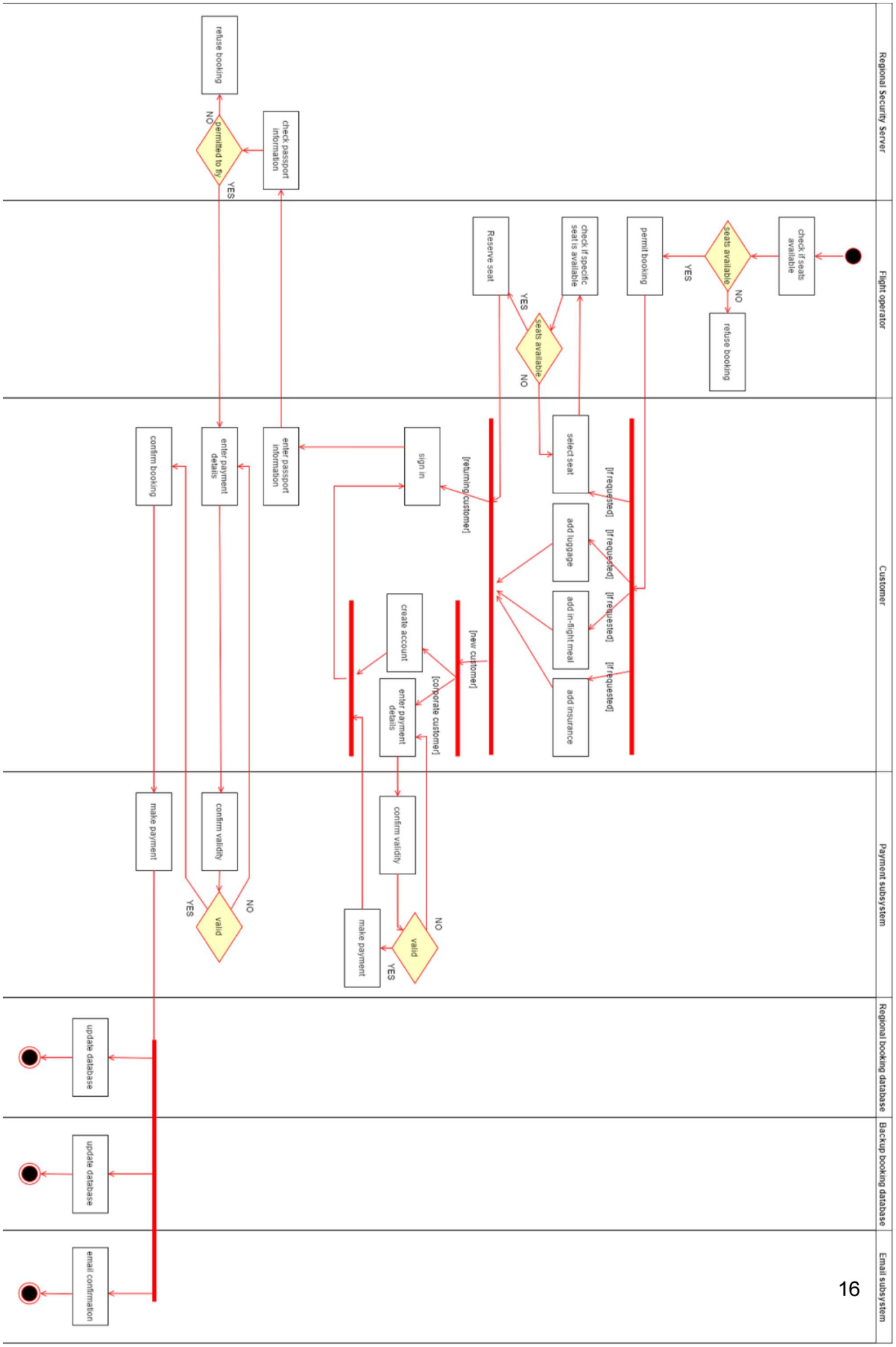
information is checked and processed by the Payment Subsystem. The Email subsystem sends an email confirmation the cancellation and any charges made. Moreover, the bookings and accounts are updated to reflect the cancellation in the Regional Database and Regional Security Server. The Backup Database updates its records every hour and only with the records which were changed more than hour ago.

Scenario:
The individual account holder named Mark has chosen to cancel his Virgin airways flight from Manchester to Prague with the system. Mark has been informed that the event he was suppose to attend to celebrate his brother's wedding has been cancelled. The flight that he booked had a clause that allows the customer to get a reimbursement if they cancel 24 hours before departure which is the case with Mark as the event was planned for the following month. He has decided to cancel his booking and attempt to get a reimbursement from the Virgin. Mark was already registered on the system and had set up an account when he first purchased the ticket. Mark logs into the system and is taken to a landing page which displays the flight that he has booked, he has no other flights booked at this time. The landing page displays the options to cancel, change booking or make another booking. Mark wants to cancel his booking therefore selects cancel booking for his flight. The system checks the time that has passed from the initial confirmed booking. The flight operator subsystem checks the amount to be reimbursed to the customer depending on the terms of the flight. The payment subsystem is notified and the final balance is calculated from the reimbursement due and cancellation charges. Mark is taken to a cancellation page which displays the balance calculated from the reimbursement from the airline minus any charges incurred for cancellation cost.

The system requires Mark to confirm the cancellation by selecting the 'confirm the cancellation' button and he does this after reading the terms of his flight in which he checks the reimbursement displayed on the screen is correct according to the conditions with the airline. He is annoyed that he has to pay a cancellation fee however he accepts this fact and continues with the cancellation process. The customer is cancelling the flight therefore the confirm the cancellation button is selected. The flight operator subsystem is updated and the pre-booked seat is made available. The database and Backup database and Regional Database are updated. The flight booking remains in the databases for a record however the booking is now cancelled so a marker is placed against the booking. All the information on the R.S.S. remains on the system for a period of two years. The email subsystem sends an email to the Mark's email account to notify them of the cancellation details along with all the financial information and updates. Mark is then redirected to the landing page of his account which has now been updated. He doesn't have any other bookings so 'no current bookings' is displayed. He is prompted to make a new booking for future travel and he considers booking a holiday for his mother as a Christmas present so starts searching for possible flights.

Activity Diagram

The next page presents an activity diagram detailing the process for the first documented use case, namely booking a flight for an individual. This is likely to be a common use case seen in the real system and demonstrates how the system interacts with its related sub-systems.


Regional Security Server
Flight operator
Customer
Payment subsystem
Regional booking database
Backup booking database
Email subsystem
check if seats available
seats available
YES
NO
refuse booking
permit booking
Reserve seat
check if specific seat is available
seats available
YES
NO
select seat
[if requested]
add luggage
[if requested]
add in-flight meal
[if requested]
add insurance
[if requested]
[returning customer]
sign in
[new customer]
create account
[corporate customer]
enter payment details
confirm validity
valid
YES
NO
make payment
update database
update database
email confirmation
enter passport information
check passport information
permitted to fly
NO
YES
refuse booking
enter payment details
confirm validity
valid
YES
NO
make payment
update database
confirm booking

Class Diagram

Noun/Verb analysis

Noun/verb analysis was applied to the requirements document to determine possible classes, class attributes and methods for the system. Each noun in the requirements document was considered for its potential as a class. Many possibilities were discarded, either for being redundant, too vague, beyond the scope of the system or more suited to being an attribute of a class. Nouns which represented whole components, for which the internal function is beyond the scope of the system are labelled as components - for example, data is stored in and retrieved from various databases which can be synced with one another, however the specifics of how this occurs is not a concern of the booking system The discarded list was then re-examined to determine possible attributes for the possible classes. The results of the noun analysis are shown below:

Candidate class	**Use**		**Candidate Class**	**Use**
flight	Class		additional baggage	Class (sub-class of add-on)
user account	Class		large item	attribute (additional baggage)
passenger	Class		type of baggage	attribute (additional baggage)
booking	Class		preselected seat	Class (sub-class of add-on)
fraudulent booking			insurance	Class (sub-class of add-on)
security			insurance company	attribute (insurance)
legal flight status	attribute (Passenger)		duplicate booking	
passport information	attribute (Passenger)		provisional booking	attribute (booking)
country of departure	attribute (Flight)		booking time	attribute (booking)
visa requirements	attribute (Flight)		booking database	Component
destination country	attribute (Flight)		total to be paid	Class
instructions			price breakdown	attribute (Total price)

visa	attribute (Flight)		base price	attribute (Total price)
confirmation	Class		Payment	Class
flight updates	Class		corporate user account	Class
confirmation emails			non corporate user account	Class
flight operator	Class		debit/credit card	attribute (Payment)
email subsystem			paypal	attribute (Payment)
flight changes			bank transfer	attribute (Payment)
add ons	Class		payment system	
inflight meal	Class (sub-class of add-on)		payment request	
dietary requirements	attribute (in-flight meal)		additional baggage	Class (sub-class of add-on)
hotel booking system			Package deal	Attribute (Booking)
recommender system			regional database	Component
unwanted parties			back-up database	Component
nationality of user	attribute (Passenger)		load balancer	Component
home region of flight operator	attribute (Flight operator)		server	Component
booking reference	attribute (Booking)		wait time	
email system	outside scope		logged in user	attribute (User account)
user email address	attribute (User account)		provisional booking	attribute (Booking)
booking specification			availability of add ons	attribute (Add-on)

bank account details	attribute (Corporate user account)		system bottleneck	

Verb analysis

Each verb and verb phrase in the requirements is then analysed to determine possible actions that classes may provide

Verb	Possible Class method	Verb	Possible class method
browse	outside scope	send passport information	Flight Booking class
sign in	outside scope	alert to visa requirements	Flight Booking class
register	outside scope	confirm with flight operator	Flight booking class
book	too broad	request to email subsystem	Confirmation class
check signed in	User account class	push flight changes	Update class
purchase a flight	too broad	determine users on updated flight	Update class
enter passenger information	Passenger class	contact relevant users	Update class
transfer user	N/A	check if addon available	Add-on class
proceed with booking	N/A	specify dietary requirements	In flight meal class
ensure legal flight status	Passenger class	determine cost	Add-on class
contact relevant users	Update class	add travel insurance	booking class
check if addon available	Add-on class	request booking with flight operator	booking class
specify dietary requirements	In flight meal class	cancel booking	booking class?
determine cost	Add-on class	determine total to be	total price class

		paid	
select baggage type	Baggage class	determine preferred payment method	payment class
check available seats	Preselected seat class	request to payment sub system	payment class
Display eligible seats	Preselected seat class	Determine data storage location	booking class
relay selection to flight operator	Preselected seat class	generate booking reference	booking class
		sync databases	Part of database component

Responsibility Analysis using CRC cards:
Using noun/verb analysis to determine potential classes, CRC cards were then created to model potential classes and how they would collaborate with the rest of the system. Collaborators in **bold** are other subsystems or third parties. Responsibilities in *italics* are likely to be methods rather than attributes.

Flight	
Flight number Departure airport Arrival Airport Departure time Travel date Arrival time Flight operator Ticket class Baggage allowance Ticket price Visa requirements Add-ons available	Flight operator Booking Add on **Flight Operator** **Security Service**

Passenger	
Adult or child Passenger name DOB Nationality Passport number Passport expiry *Legal flight status*	Flight Booking **Security Service**

Flight Booking (subclass of booking)	
Passengers Flights Add-ons *Visa needed*	Passenger Flight Add-on **Flight operator**

Update	
Flight Changes made *Users to contact*	Booking **Flight Operator** **Email**

Confirmation to User	
User account Booking *Send confirmation*	Booking User Account **Email**

User Account	
User identifier Contact email Previous bookings *Check signed in*	Booking **User Database** **Recommender**

Add-on (abstract)	
Cost *Available on flight* *Add to booking*	Flight Flight Booking Total price **Flight operator**

Booking	
Booking reference User account Time of confirmation *Booking confirmed* *Confirm with flight operator* *Add to database* *Update booking*	User account Flight Booking Confirmation Update **Flight operator** **Booking database**

Additional Bag (subclass of Add-on)	
Large/heavy item Sports gear	Add-on Total Price **Flight operator**

Flight Operator Details	
Name Country of registration Contact details Payment details	Flight

Insurance (subclass of Add-on)	
Insurance Company Insurance Policy	Add on Total price **Insurance**

Payment	
User Booking Debit card Credit card Paypal Bank transfer *Request payment*	Total price User Account Booking **Payment system**

Total Price	
Base price Add-on costs *Total cost*	Flight Add-on Payment

In-flight meal (subclass of Add-on)	
Dietary Requirements	Add-on Total Price **Flight operator**

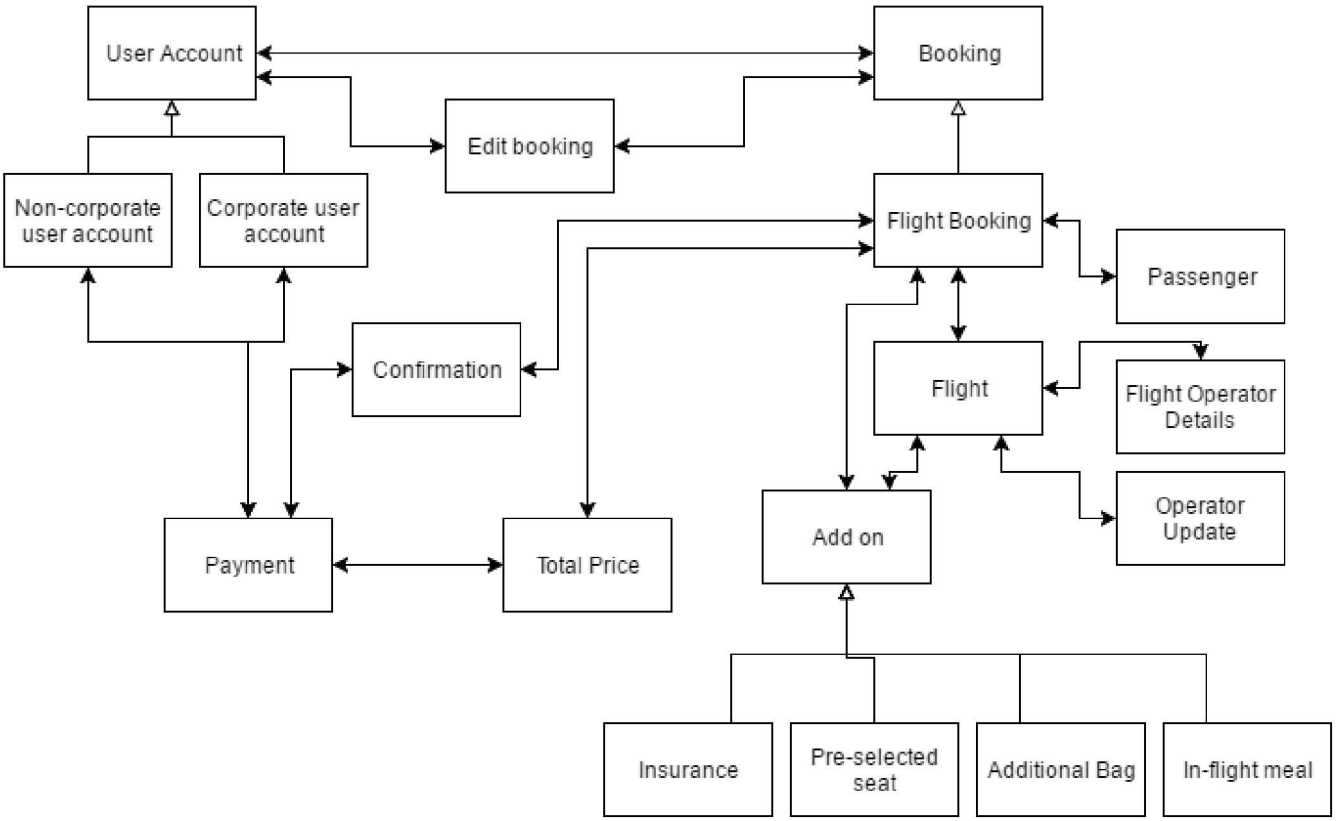
Corporate user account (subclass of User account)	
Contact address Company name Bank account details	User account

Pre-selected Seat (subclass of Add-on)	
Seat number *Get available seats*	Add-on Total Price **Flight operator**

Non-corporate user account (subclass of user account)	
	User account

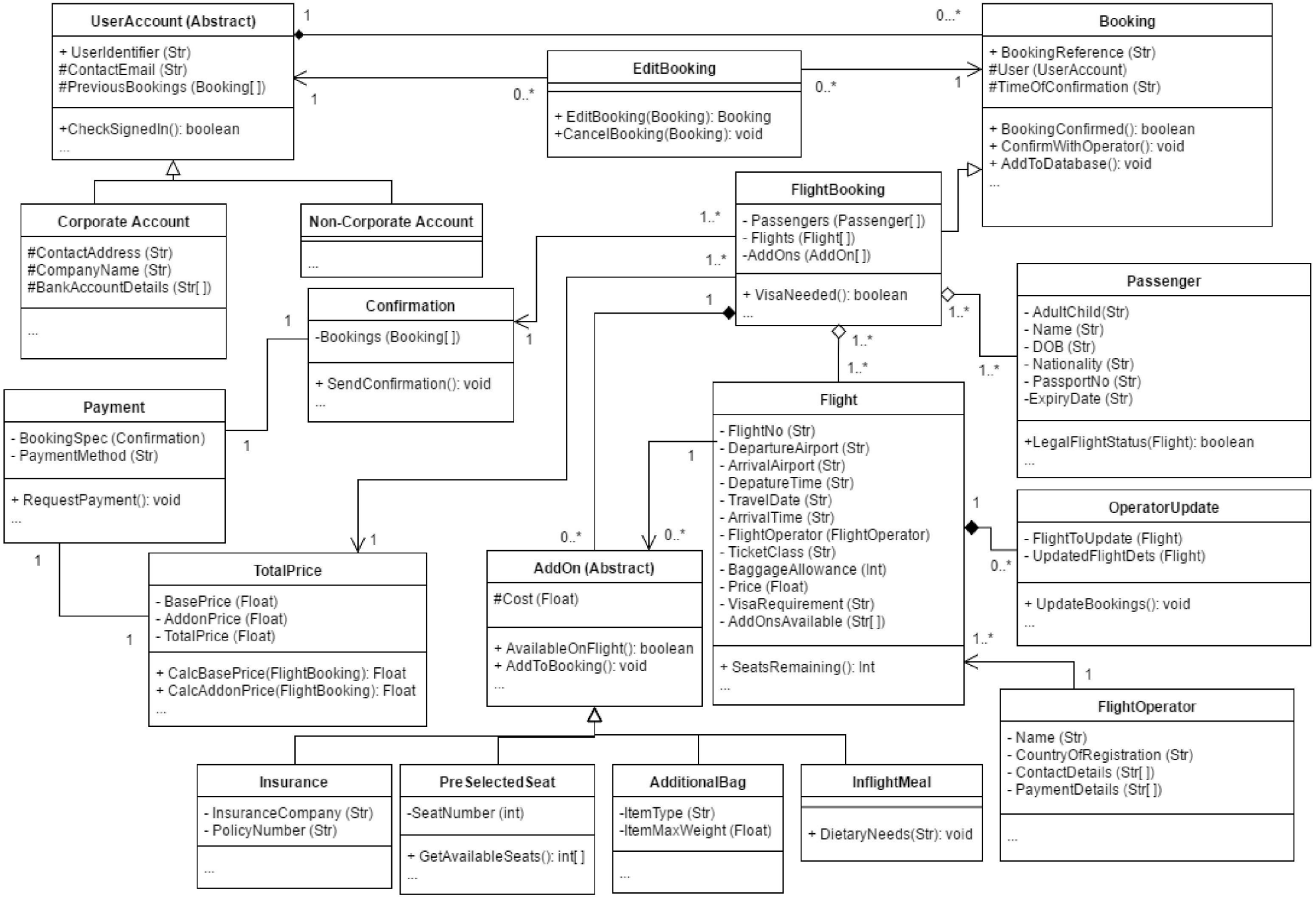
Edit booking	
Cancel booking *Change booking*	User account Booking Payment **Booking Database**

The CRC cards were used, along with the noun/verb analysis to create a first cut class diagram, showing how the classes would link together, this is presented below:



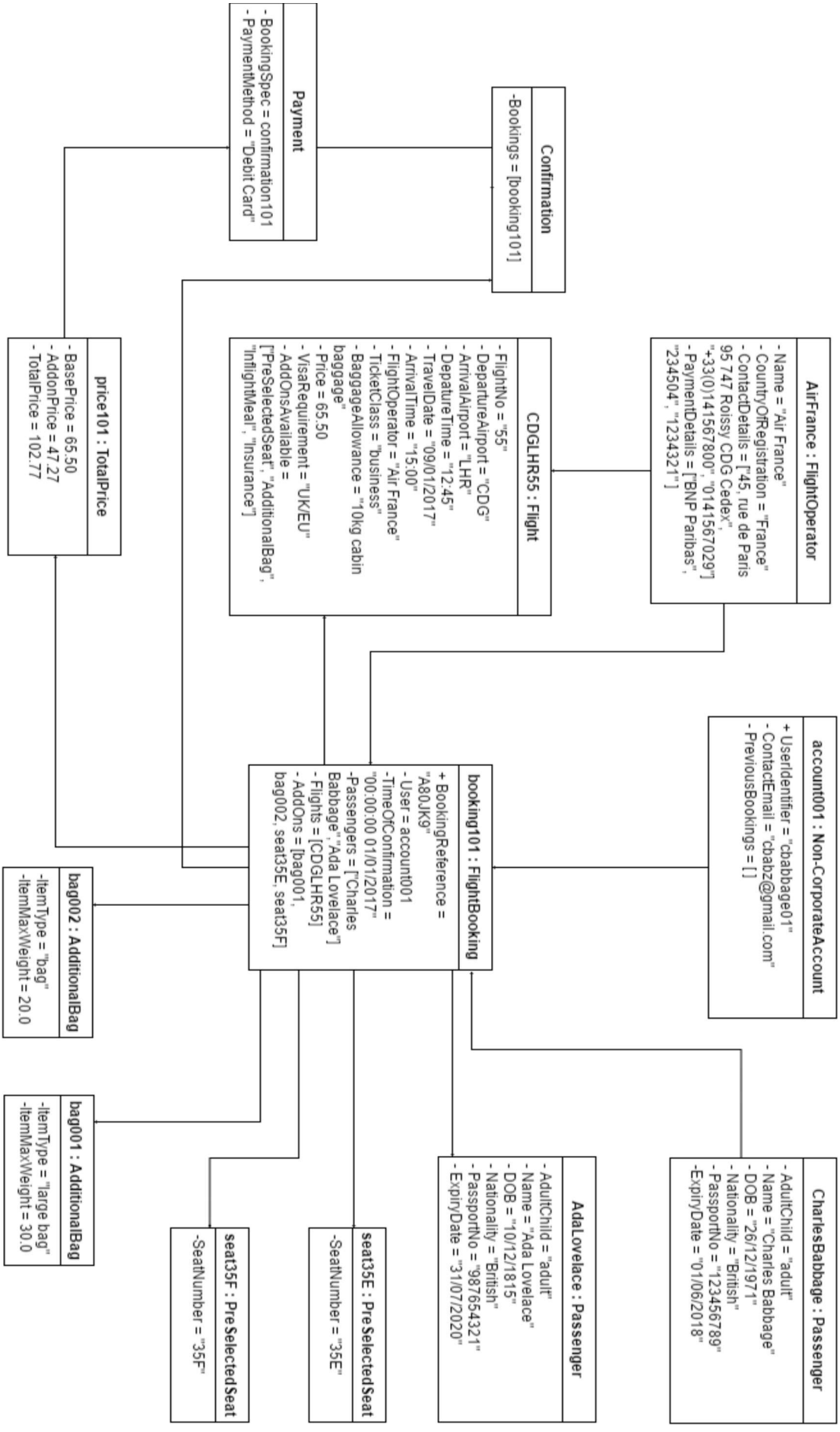
From this first cut diagram, the collaborators of each class can be easily determined. Using this diagram, a full class diagram was created, detailing the likely contents of each class as well as the structure of the design. This is presented on the next page.

*ellipses have been used to indicate additional methods such as getters and setters for attributes which would also be present but which have been excluded for readability of the diagram



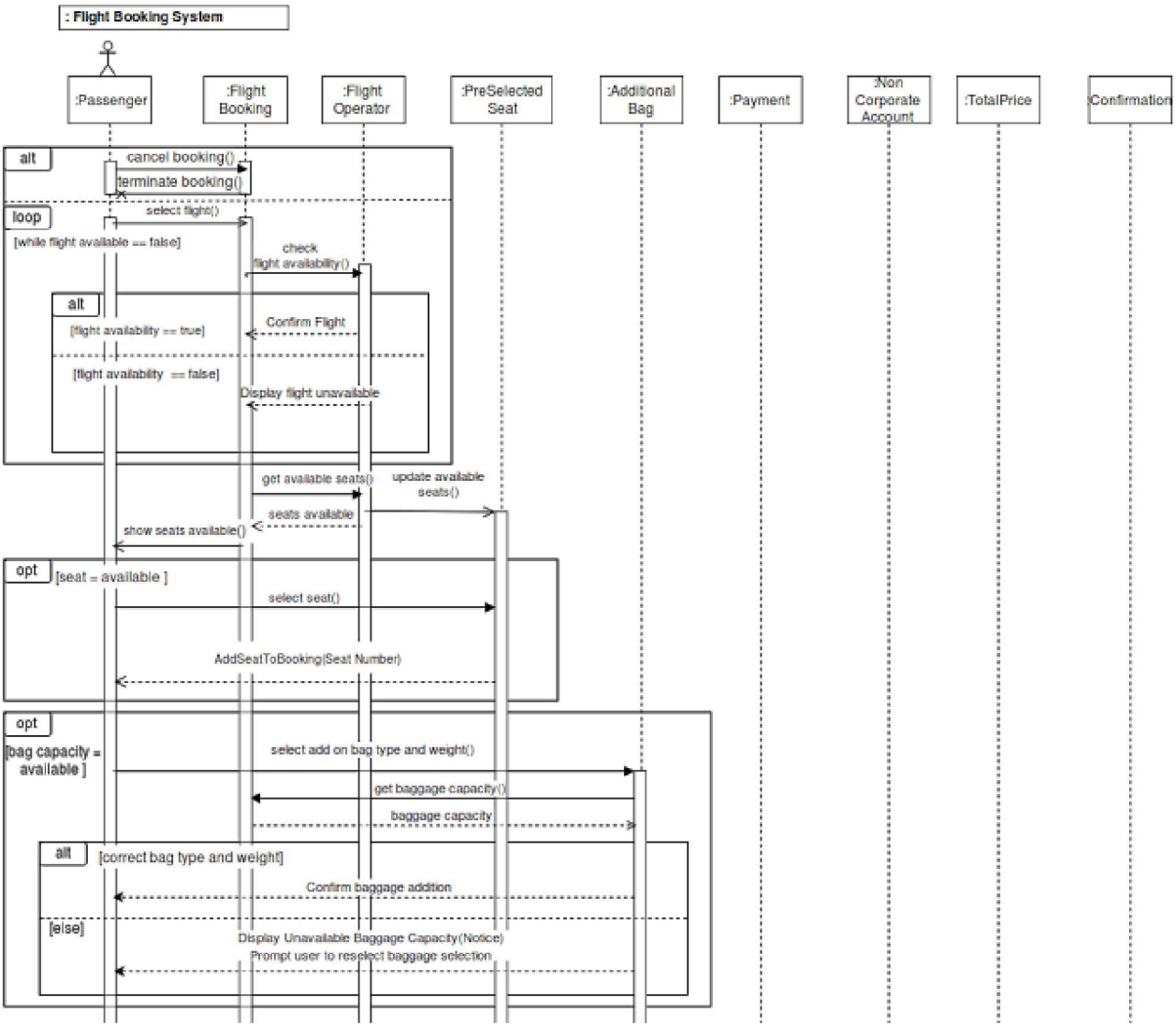
Object Diagram

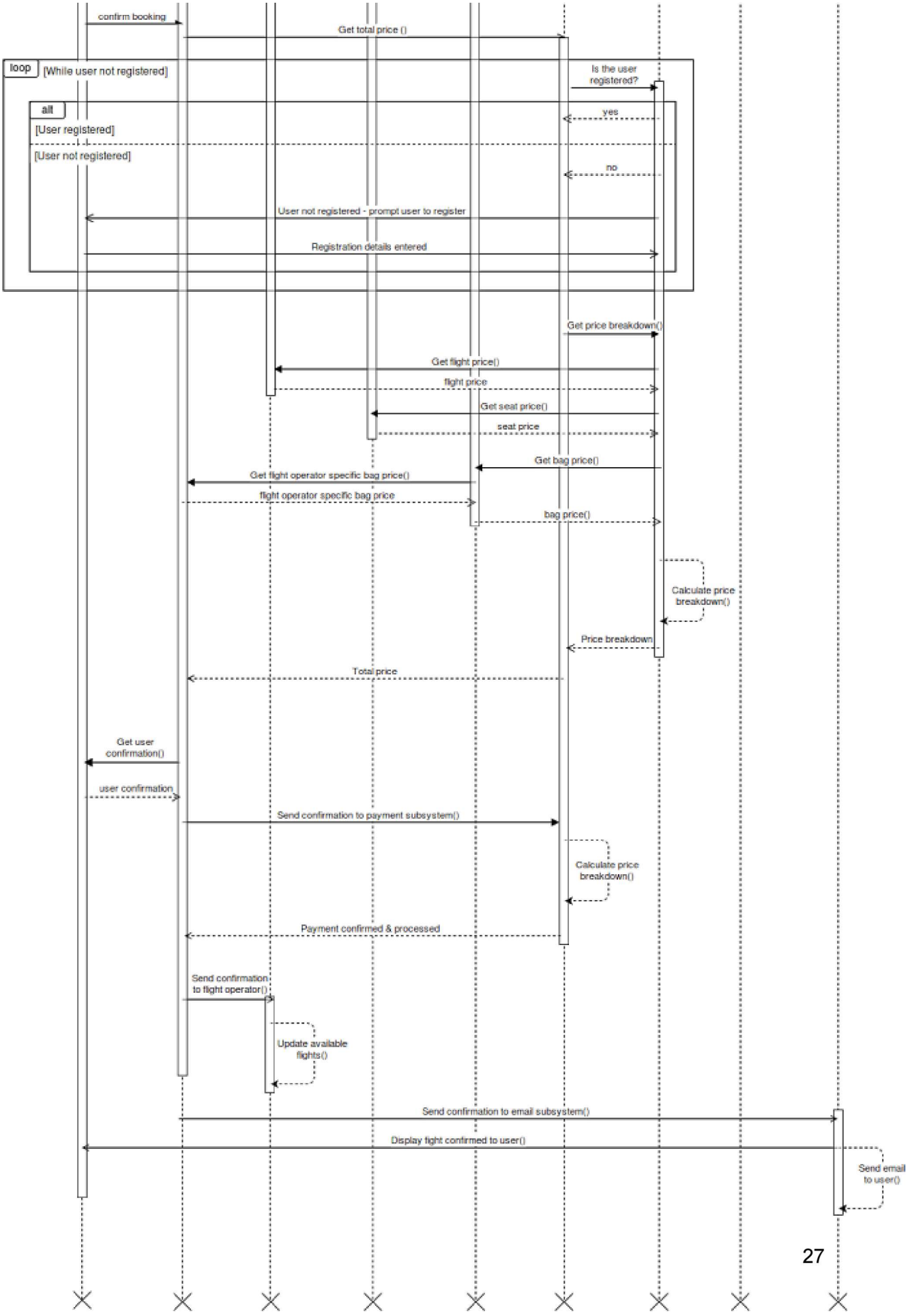
From the class diagram, an object diagram was created, showing a snapshot of the contents of the system's attributes during operation:



Sequence Diagram

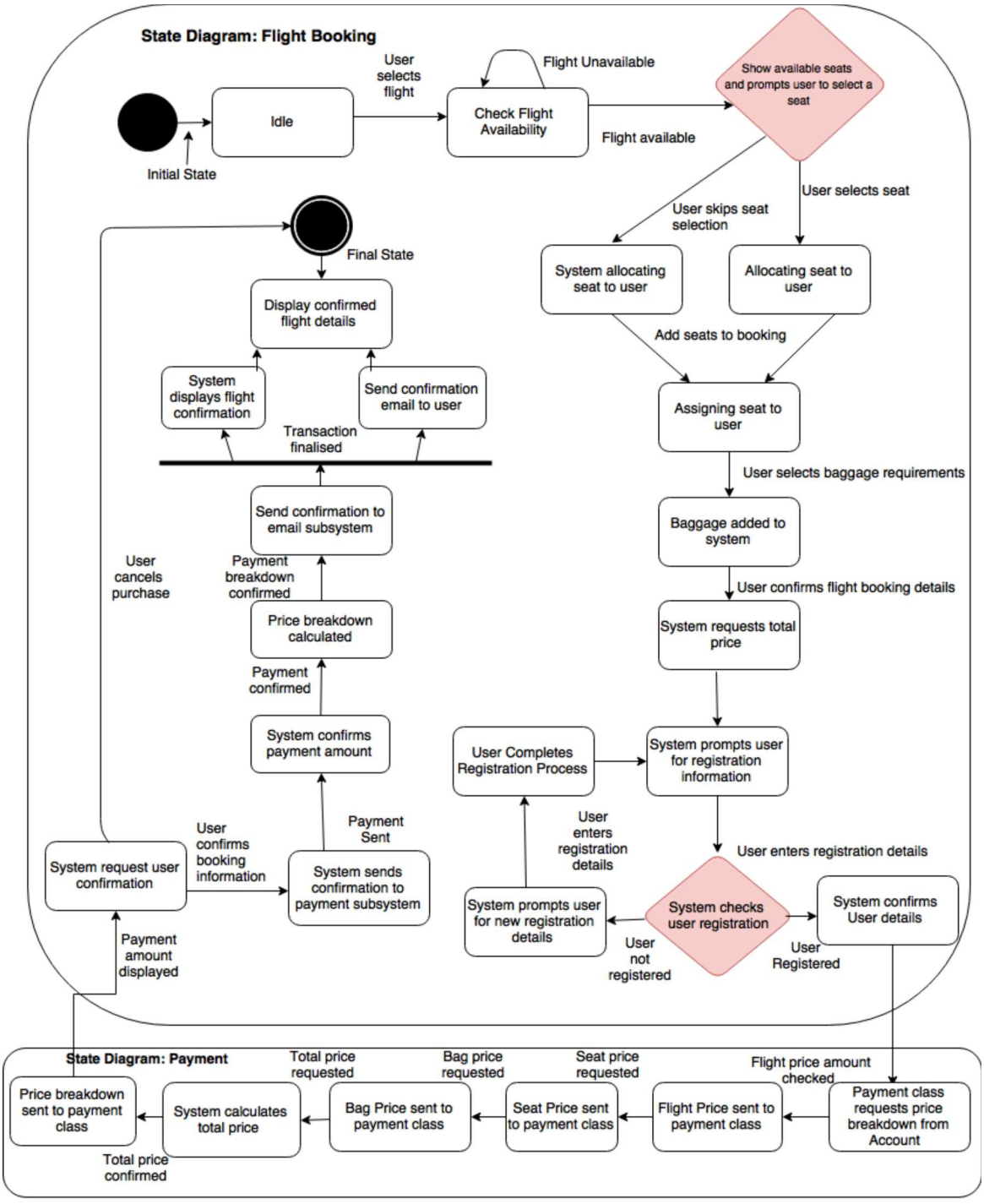
This section shows a Sequence Diagram for the scenario of booking a flight. The sequence starts with the Passenger actor either cancelling or selecting her flight. The Passenger then enters a loop until a valid flight is selected or it is cancelled. The Passenger is then presented with options to pre book a seat and add extra luggage space. Once the extra options or chosen (or not) the system calculates the total price and ensures the Passenger is registered during this process. Following this, the booking is confirmed and is displayed to the user and via an email confirmation.




confirm booking
Get total price ()
loop
[While user not registered]
Is the user registered?
alt
yes
[User registered]
[User not registered]
no
User not registered - prompt user to register
Registration details entered
Get price breakdown()
Get flight price()
flight price
Get seat price()
seat price
Get bag price()
Get flight operator specific bag price()
flight operator specific bag price
bag price()
Calculate price breakdown()
Price breakdown
Total price
Get user confirmation()
user confirmation
Send confirmation to payment subsystem()
Calculate price breakdown()
Payment confirmed & processed
Send confirmation to flight operator()
Update available flights()
Send confirmation to email subsystem()
Display fight confirmed to user()
Send email to user()

State Diagram

The state diagram shows the specific components and objects of a system which, in the case of the current system, consists of the flight booking subsystem. It has been selected as it is one of the main components within the full system. The flight booking component interacts with other components within the system, especially the payment component. Therefore the payment component has also been documented at the bottom of the diagram to display greater insight into how the components interact with each other.

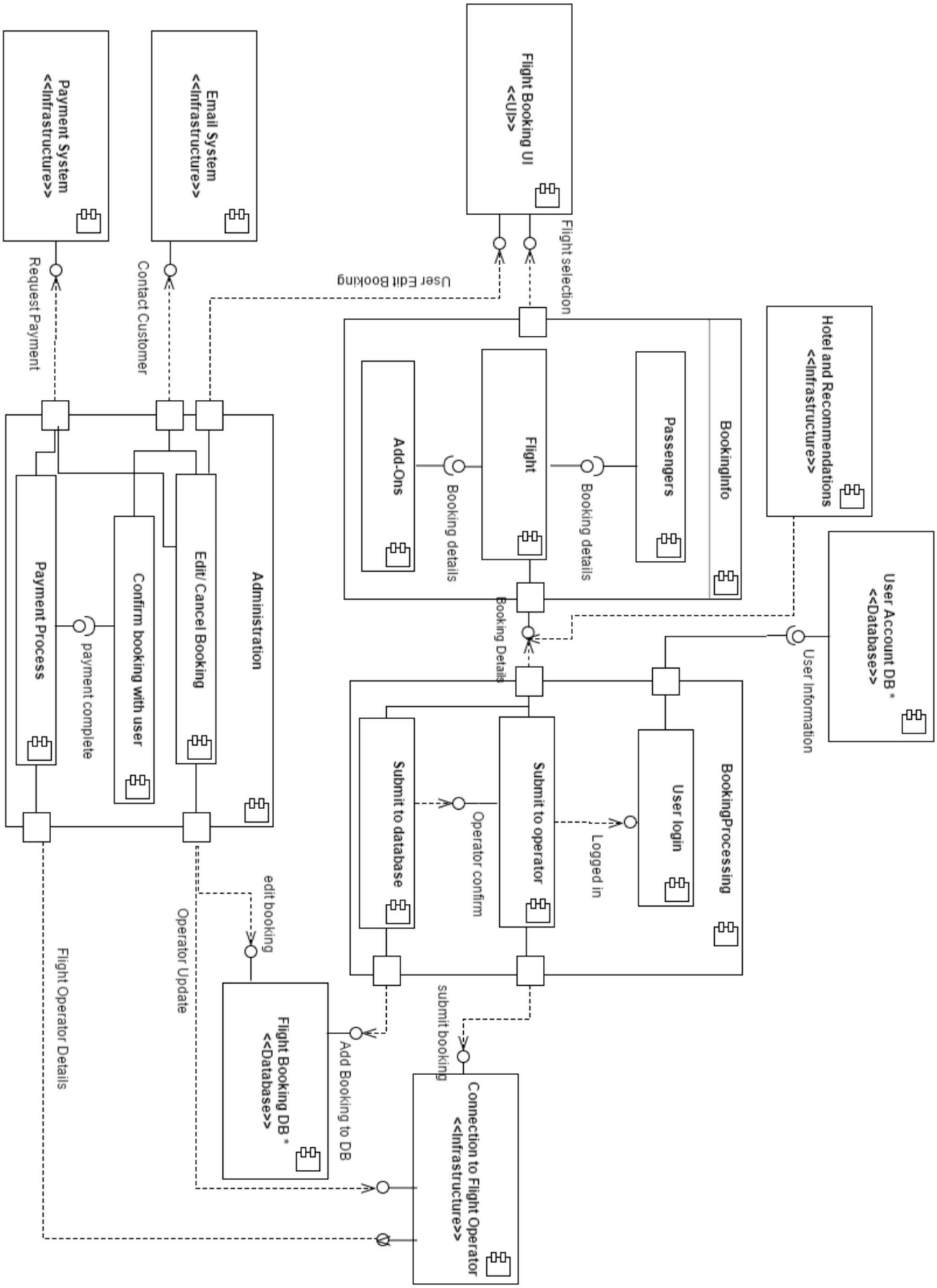


Architecture Designs

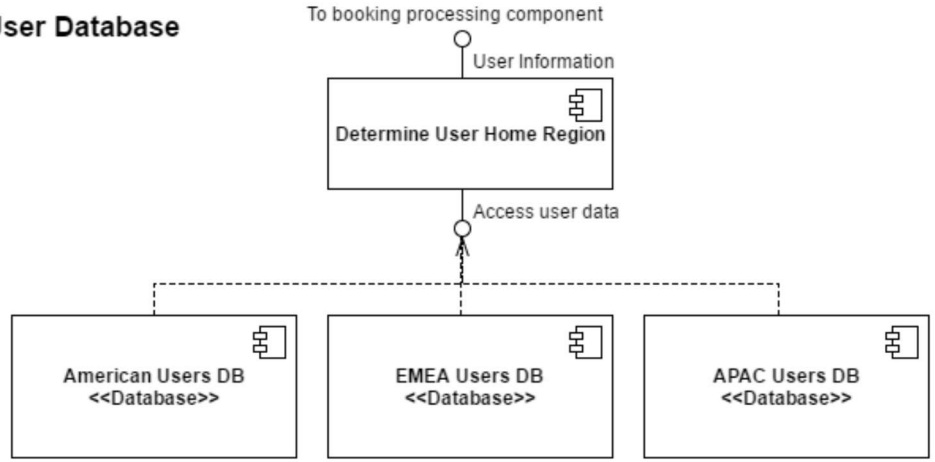
Two possible designs are put forward for the system. Both follow the strategy of componentization using tradeoff analysis to determine a suitable architecture. Component and deployment diagrams are provided for each of the two architectures as well as a brief discussion of the rational behind the proposed design and the tradeoffs considered.

Design One:

Component Diagram
The component diagram for this design consists of two parts, the first demonstrates the componentization of the majority of the system. The second illustrates the setup of the database systems, both for flight booking details and user account details.


Flight Booking UI
<<UI>>
Payment System
<<Infrastructure>>
Email System
<<Infrastructure>>
Hotel and Recommendations
<<Infrastructure>>
User Account DB *
<<Database>>
Flight selection
User Edit Booking
Request Payment
Contact Customer
BookingInfo
Add-Ons
Flight
Passengers
Booking details
Booking details
Booking Details
User Information
Administration
Payment Process
Confirm booking with user
Edit/ Cancel Booking
payment complete
BookingProcessing
Submit to database
Submit to operator
User login
Operator confirm
Logged in
edit booking
Operator Update
Flight Operator Details
Flight Booking DB *
<<Database>>
Add Booking to DB
submit booking
Connection to Flight Operator
<<Infrastructure>>

User Database

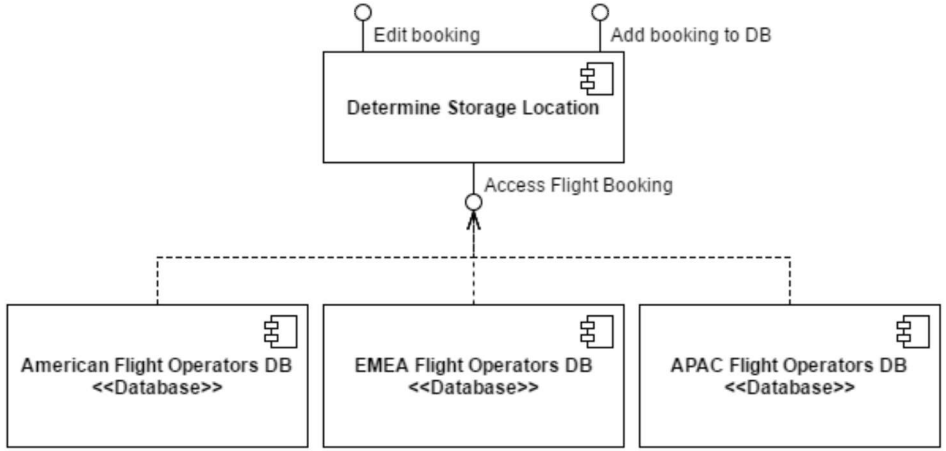


Interface: Access user data

Interface allows user information to be accessed from the correct regional database. It also allows booking references to be added to a user account

For security reasons, user information must be stored on a database in the region corresponding to the user's nationality. This means that a user's data will be subject to the privacy laws of their home region and data cannot be used legally by other parties

Flight Booking Database



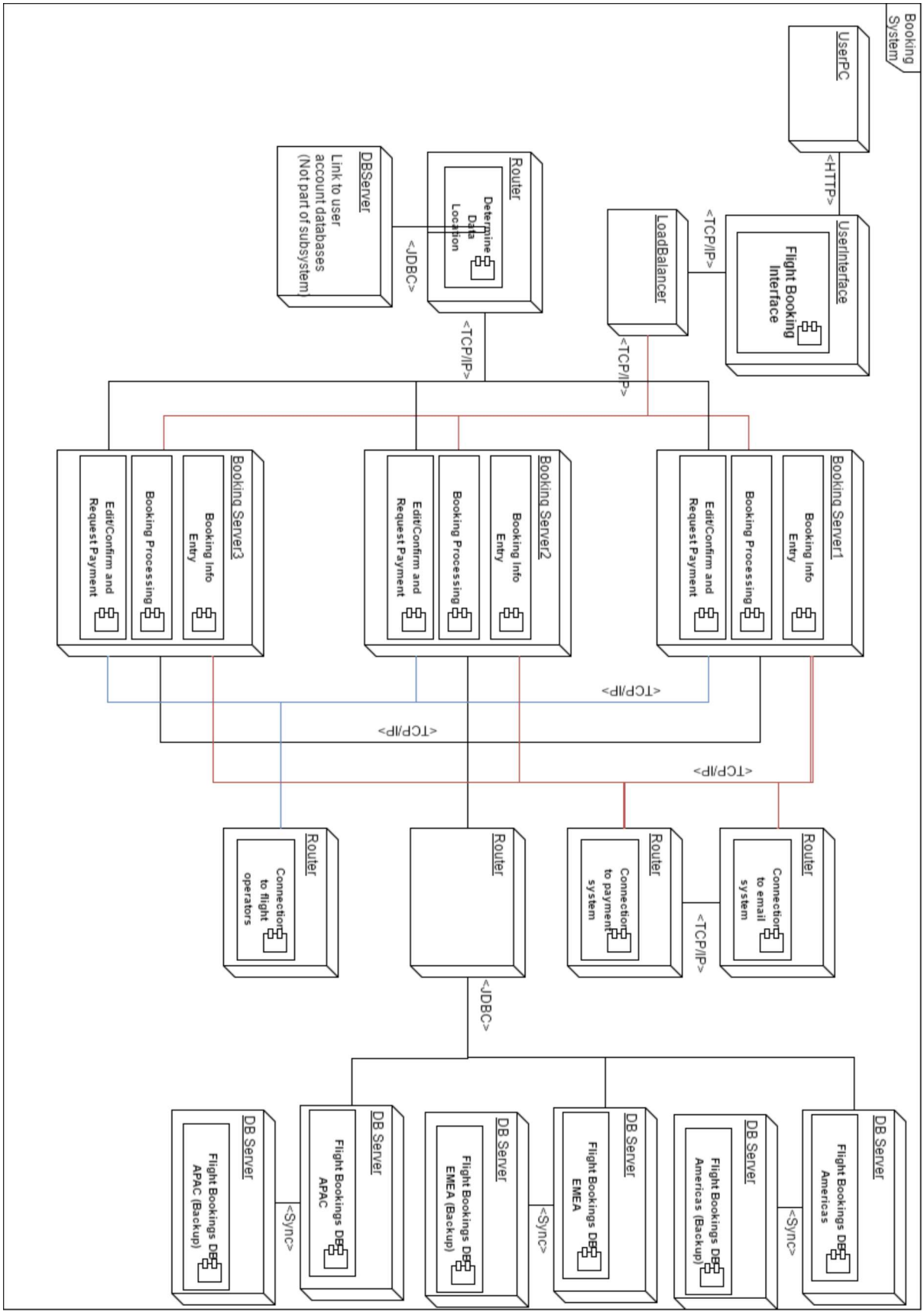
Interface: Access Flight Booking

Interface allows flight information to be stored in the correct regional database. It also allows a booking related to a specified booking reference to be edited either due to an operator update or by the user

For security reasons, flight booking information must be stored on a database in the region in which the flight operator is registered. This means that a user's data will be subject to the privacy laws of the operator's home region and data cannot be used legally by other parties

* diagram does not include back-up systems which will be explained in the deployment diagram

Deployment Diagram



Tradeoff Analysis
In designing an architecture for the flight booking subsystem, the non-functional requirements of the system were considered. The system ideally needs to be secure, efficient, scalable, inexpensive and easily maintainable however these aspects can often create conflicts in the design process and as such, trade-offs must be made when designing a real system.

Component Diagram: In choosing components for the system architecture, the aim was to reduce traffic between components to promote efficiency and reduce coupling. The system relies on interaction with several other subsystems which must also be included.
It was decided to group all the information collection from the user into one component containing the relevant sub-components. This allows all the booking information to be transferred to the bookingProcess component in one package. It also means that if the method of collecting the required information is altered, only the bookingInfo component needs to be modified, hence providing a system with lower coupling.
The bookingProcess component is then responsible for the actions of actually making the booking, both with the flight operator and with the systems own databases. This means that this component is not directly interacted with by the user but instead is responsible for transferring the required details to the necessary parties (Flight operator and databases). There were several other processes such as requesting payment and sending confirmations to the client which were not directly related to creating the physical booking. These tasks were all considered to be administration tasks, tasks that either the system or the user may need to undertake after the booking had been created. In terms of accessing the payment system and the email system, it seemed sensible to add the functionality of requesting payment and requesting confirmation be sent to the user in this component. The difficulty was deciding where to add the functionality to edit/cancel a booking, both on the user side and that of the flight operator. Given the need to interact with the user as well as there being potential for additional payments, it was decided to add this to the administration component to reduce coupling and improve maintainability however there would also be an argument for adding this functionality to the bookingProcess component due to its role as intermediary between the user and the flight operator/database. Providing it as a component on its own didn't seem to be an improvement on either of the ideas suggested previously.

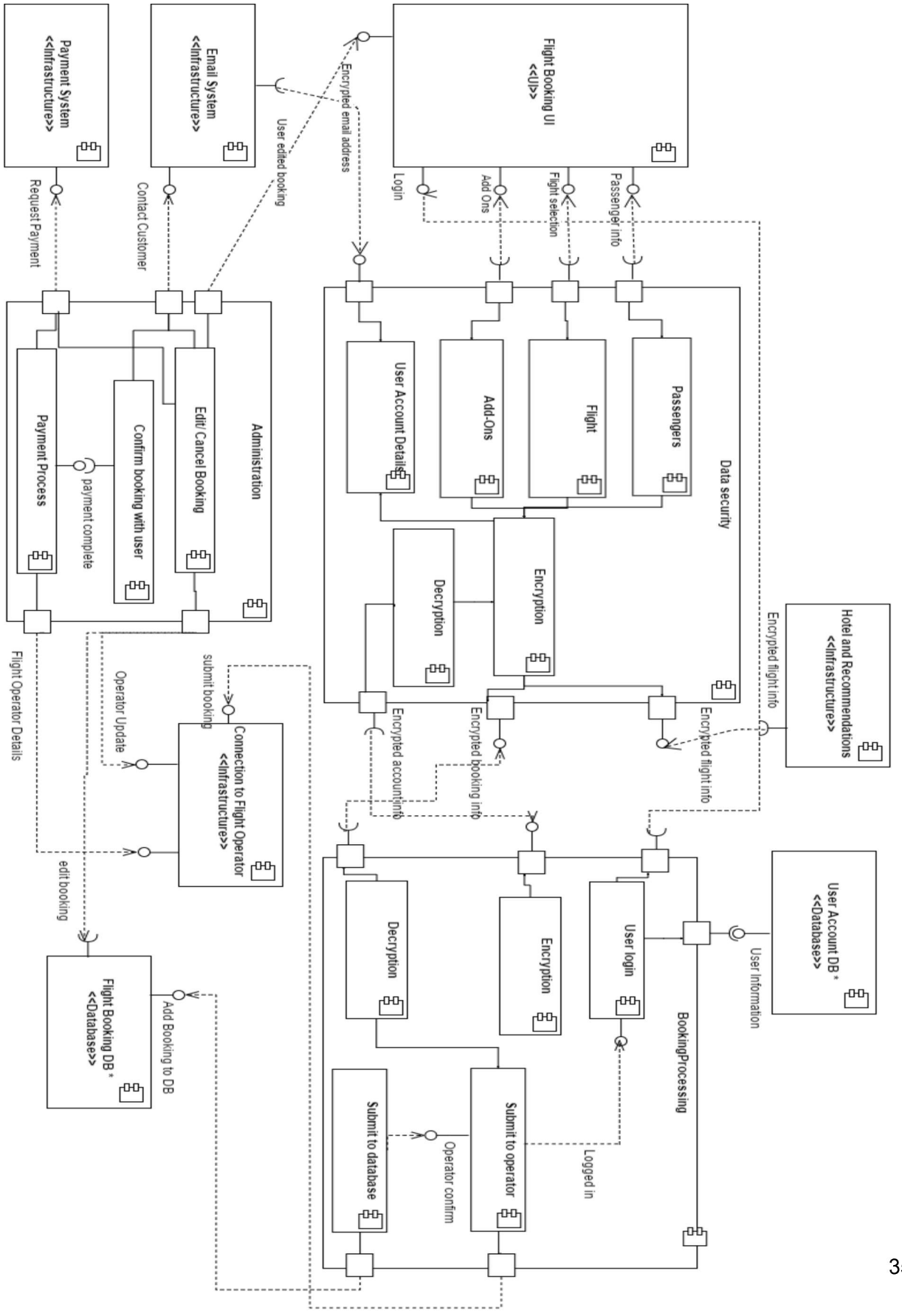
Deployment Diagram: For this flight booking subsystem, personal information is provided by users and as such security must be a priority, in particular secure data storage. This is likely to increase the cost of the project however considering the potential cost of legal proceedings resulting from improper use of client's data, the additional cost of storing data on several regional databases instead of all together is deemed worth the compromise on cost. The presence of back-ups for each database which regularly sync with each of the primary databases improves the maintainability of the system. A database can be taken offline for maintenance without needing to back up data and if a database malfunctions, the back-up can take over primary function and be used to restore the main database. Again this increases cost and the loss in revenue caused by the system being offline must be considered in the design. The system also utilises 3 identical booking servers, each of which

contain all 3 major components of the booking process. This means that, through use of a load balancer, a user can be directed to the server with the least traffic to complete their booking, making the process efficient and scalable, as additional servers can also be added. The use of multiple servers improves maintainability in that the system never has to be taken fully offline for server maintenance. However the downside of multiple servers is that there are more components that require maintaining and updates must be rolled out to all three servers.

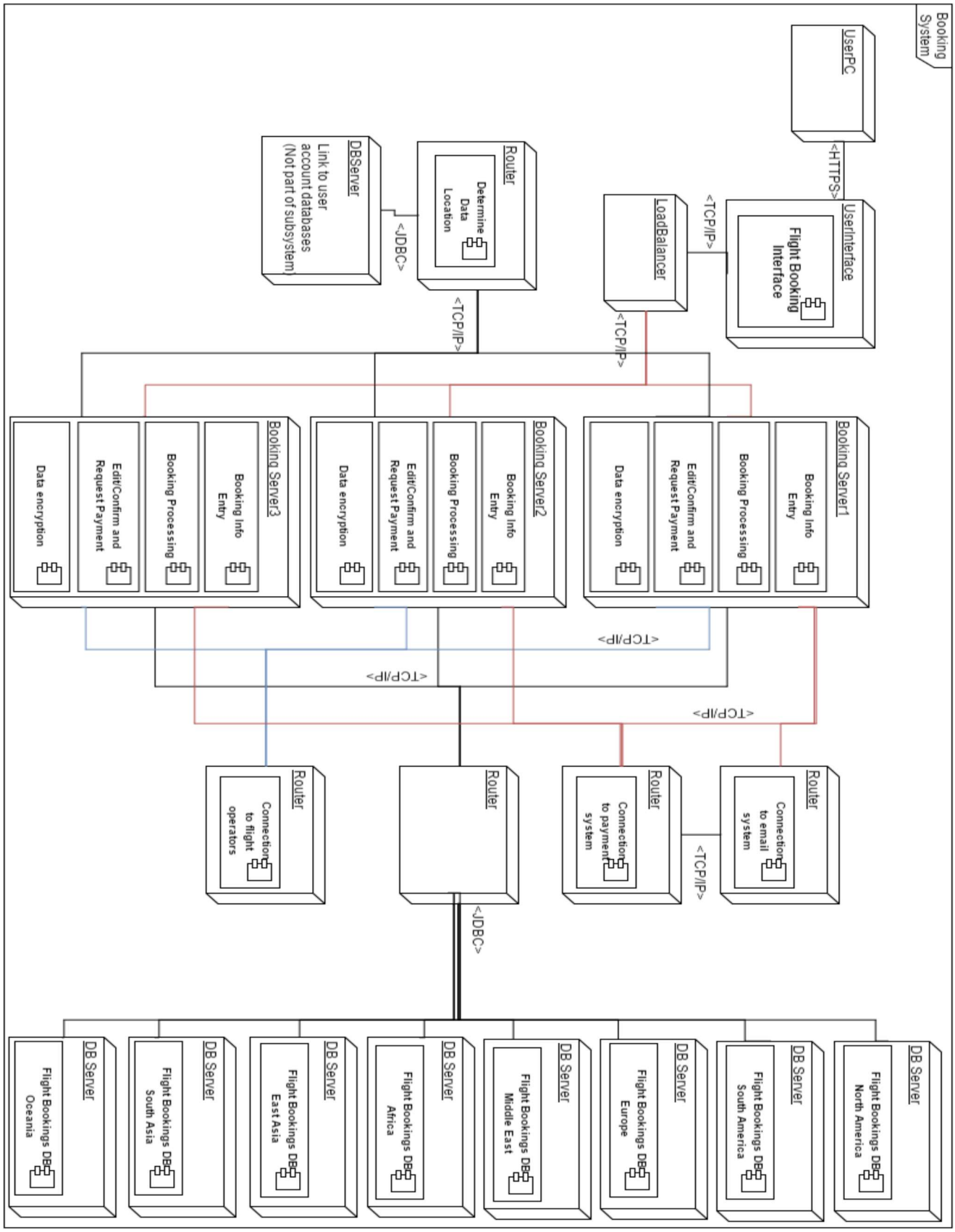
One issue with the proposed architecture is that each connected system, email, payment, flight operator etc each uses a single router despite traffic coming from up to three booking servers. As such, there is the potential for bottlenecks in the process which can reduce efficiency and availability. An alternative considered was to have separate routers for each server however this would just move the bottleneck to the router-database connection. It was decided that the addition of database systems for each server would make the system too expensive and could cause maintainability issues as well as making the synchronisation process more complicated and prone to error.

Design Two:

Component Diagram



Deployment Diagram



Tradeoff Analysis

Component Diagram: Similarly to the first architecture design, steps were taken to reduce the overall need for traffic between components and reduction of coupling. However in this instance there was a greater emphasis placed on data security and the security of communications between components, and thus a more centralised data component was introduced that include inner components to facilitate encryption of information. Centralisation of the data components for flight data and account data makes it more efficient to protect and offers fewer opportunities to illegally access this data. Individual elements of the user inputs to the UI are transferred separately to the central component, such that if one element of the information is compromised the rest may remain secure. Thus there has been some trade-off of efficiency and security.
The encryption process occurs within this central component, and information must be decrypted by any receiving components before the content can be analysed. While encryption is centralised, it makes sense that decryption must be localised to each component that must access the flight, passenger, and account information. This adds additional complexity to the existing components and makes maintenance of the system more difficult, however it ensures that information is secure as it is passed between elements of the system and other subsystems which is particularly important as regards account details such as name and email address, and passenger details such as passport number and date of birth.

Deployment Diagram: The two designs put forward for the deployment of the system are highly similar, but differ in some key and important features. Efficiency and security of user connections to regional servers is enhanced by increasing the divisions of the previous regions (Americas, APAC, EMEA) into more localised servers- a single American servers becomes a North and South American server, etc. To reduce the cost of running these extra servers, no backup servers are included in this design, and thus there has been a trade off in ease of maintenance and cost/performance/security/accessibility. Performance will increase due to the individual regional servers handling less traffic due to their greater specificity, will also reducing the communication distance between users and their nearest server.
It is not sufficient for security to focus on data storage alone to ensure a high degree of security. To facilitate secure communication and data transfer additional encryption and decryption modules have been introduced to each booking server. Again, security is prioritised over ease of maintenance, as an alternative option would be to encrypt information via an external server that receives and transmits data to all the booking servers. However, this set-up would generate a bottleneck within the network, in addition to that represented by the single router, and so delegation of the encryption-decryption function to each individual servers seems like the more sensible and cost-effective choice. In addition, an external server would necessitate one more transfer of unencrypted data between nodes, and would thereby increase the opportunity for interception of sensitive information.