Cruise Booking Software: Software Design Document

Ivan Novasak

Southern New Hampshire University

IT 632: Software Design and Modeling

Venkatesh Baglodi

2024 December 15

Cruise Booking Software: Software Design Document

The purpose of this document is to fully guide a development team in the creation of a cruise booking software application. The first sections concern the purpose and desired functionality of the application, the advantages and disadvantages of a web app vs a desktop app, and an example of booking information for a cruise. The second section shows six use cases the software could have, along with diagrams to visualise these cases and the involved actors. The third section discusses two examples of sequence diagrams that show how functions in the program could work over the course of time. This section also covers UML class diagrams, which will show a way of turning the use cases into software objects and methods that could be programmed. Finally, the fourth and last section concerns the future, with the next steps for development for the product to be ready for implementation.

# Software System Overview

**Purpose and Function**

According to Cruise Critic, booking a cruise for travellers involves a minimum of the following elements (Cruise Critic, 2022):

* Cruise ship name
* Cruise line (the company running the cruise ship)
* Voyage name (if applicable)
* Specific cruise amenities (if additional pricing is required in the initial booking)
* Cabin type and price
* The travellers themselves/their contact information
* Booking agents
* Locations the cruise may dock at (also known as ports of call) if any
* Dates and locations the cruise will embark/disembark
* Identification information for all travellers (passport number, country, expiry date)
* Payment methods for a deposit, final payment, and port/country taxes (credit card number, type, expiry date)
* Travel insurance (not strictly required but highly recommended)

A traveller may book a cruise via a third-party travel agent or directly with the cruise line (Cruise Critic, 2022). Either way, a traveller’s booking arrangement will need to be handled and verified by an agent (Cruise Critic, 2022). Oftentimes, the traveller will need to also have transportation arrangements between their origin city to the embarkation and disembarkation ports, which may involve an additional trip by air/train/bus or even a hotel (if the traveller(s) need to arrive in/leave from the port’s city on an earlier/later date than embarkation/disembarkation) (Cruise Critic, 2022). So, the software will need a way to take in all this information to create a booking. An example of the information needed for a cruise booking is as follows (Cunard, 2024):

* Cruise ship name: Queen Mary 2
* Cruise line: Carnival plc (d/b/a Cunard Line)
* Voyage name: 185th Anniversary Voyage, 7 Nights (M517)
* Specific cruise amenities (if additional pricing is required in the initial booking): On Board Photography
* Cabin type and price: Britannia Balcony (sheltered) / $2,396.53 USD (quoted *per person assuming double occupancy*)
* The travellers themselves/their contact information:
  + Ivan Novasak / 2685 Aramingo Ave, Philadelphia, PA 19125, USA / +1 401 400 2333
* Booking agent: [Cunard Lines agent]
* Locations the cruise may dock at (also known as ports of call): Westbound Transatlantic - no ports of call between embarkation and disembarkation
* Dates and locations the cruise will embark/disembark
  + Embarkation date and location: 24 June 2025 / Ocean Cruise Terminal (Berth 46), Dock Gate 4, Cunard Rd, Southampton SO14 3QN, England, UK [address from Google Maps - verify with Cunard during actual booking]
  + Disembarkation date: 1 July 2025 / Brooklyn Cruise Port, 210 Clinton Wharf, Brooklyn, NY 11231, USA [address from Google Maps - verify with Cunard Line during actual booking]
* Identification information for traveller 1 of 1:
  + Passport country: USA
  + Passport number: A#########
  + Expiry date: 30 May 2033
* Payment methods for a deposit, final payment, and port/country taxes: #### #### #### #### / Visa / 12/29 / ###
* Payment amount to Cunard Line: $4,818.96 USD
* Travel insurance: [Paid Travel Guard]

**End Users and Organisation(s)**

Based on the FAQ by Cruise Critic, the most ideal users for the cruise booking software will be travel agents, cruise lines, travellers/holidaymakers, and travel insurance agencies (if they wish to integrate their insurance products with the cruise bookings) (Cruise Critic, 2022). Other travel and travel-adjacent businesses like airlines, hotels, tour/excursion companies, etc can benefit from the software too, so the users can book as many aspects of their trip within the same software as possible.

**Operating Environment: Web vs Desktop Application**

According to Jemin Desai of Positiwise Software, there exist several advantages and disadvantages to going with a web application vs a desktop application. The advantages of a desktop app include (Desai, 2023):

* Offline access: No requirement for an Internet connection to use the software
* Better security via local data storage without the requirement that app data be transmitted online, where it can be captured or corrupted via a man-in-the middle attack or a faulty update
* More economical because the software does not have a monthly fee for maintenance and web hosting; it can be installed on users’ computers
* Faster performance and more features

Disadvantages of a desktop app are (Desai, 2023):

* Higher hard disk space and other hardware requirements like RAM, GPU, a specific OS, etc due to the need to install the full app on the user’s computer
* Upgrades must be done manually via a new download and installation on each computer the software is installed
* The software can be accessed only from the computer(s) it is installed on, so if anything goes wrong with those computers, the software and its associated data is either inaccessible or lost
* More complicated programming languages required to build the app: C#, C++, Java, etc

In contrast, a web application has the following advantages (Desai, 2023):

* No download and installation required; the software runs via the web browser as it is a web service hosted on cloud servers
* Agnostic of operating system, web browser type, and Internet connection type, allowing the user to use the app and its associated data from any device with an Internet connection and web browser
* Upgrades are rolled out on the cloud server, with users not needing to upgrade anything besides their web browser (which is typically an automatic activity)
* Less system requirements such as hard disk space and RAM due to app running in the web browser and not on the user’s computer
* Easier programming languages to create the app such as Python, PHP, and JavaScript

Some disadvantages of using a web application are (Desai, 2023):

* Fast, stable Internet connection is a must - web apps typically will cease working if the Internet connection is lost, though some can support some temporary offline data storage that will re-update once the connection is restored
* Disruptions on the Internet can slow down the app’s performance
* Security issues making encryption of all transmitted data a must-have; managers must hire cyber security specialists who are up to date with the latest security threats and how to deal with them
* High monthly, quarterly, or yearly subscription rates to pay for maintenance, upgrades, web hosting even for a web app that does minimal functionality

**Software Analysis and Use Cases**

**Actors**

The actors that are involved with booking a cruise are all the following:

* The customer(s) (passengers)
* Customer service/verification staff at the travel agency/cruise line
* The bank who processes the customer’s payment
* The web server for showing the customer their cruise options and submitting the booking
* The booking engine which transmits payment and gets cruises to display to the customer
* The cruise line who runs the cruise
* The travel insurance company (assuming the passenger purchases separate travel insurance)

**Use Case Scenarios**

Tables 1 through 6 lay out various use case scenarios for a user who wants to check availability of and book or modify cruises. The layout is based on the one found in (Larman, 2004, pp 67/68). The cases to be covered are:

1. Searching for cruises (booked or non-booked)
2. Viewing information about a specific cruise
3. Booking a cruise
4. Modifying or cancelling a cruise booking
5. Requesting room service/other amenities on the cruise
6. Contact a customer services representative

**Table 1**

*Searching for a Cruise*

|  |  |
| --- | --- |
| Actor(s) | * Customer (Primary) * Cruise Web App (Supporting) * Booking Engine (Supporting) |
| Description of the Functionality | The customer should be able to search for any cruise in the system, view their own booked cruises, and filter amenities, dates, locations, duration, budget, etc. |
| Precondition(s) | The customer is logged into the web app |
| Assumption(s) | The customer wants more information about cruises or wants to view their bookings |
| Flow of Events | 1. Customer logs into the web app 2. Web app displays options for the customer 3. Customer chooses either Search for a Cruise or View My Bookings 4. Customers are then prompted to enter a search box or click tick boxes to filter on amenities choices, destinations, durations, dates, etc. They can also choose to view all cruises. 5. Customer enters their desired cruise search categories or filters (if any), and hits Submit 6. Booking engine gets cruise data and sends it to the web app 7. Web app displays the cruises for the customer to view more information about 8. System displays an indicator for which cruises the customer has booked |
| Postconditions | Listing of cruises and/or bookings is displayed, with a thumbnail picture showing the cruise line and a photo of the ship |

**Table 2**

*Viewing Information about a specific Cruise*

|  |  |
| --- | --- |
| Actor(s) | * Customer (Primary) * Cruise Web App (Supporting) * Customer Services Representative |
| Description of the Functionality | The customer should be able to get more information about a specific cruise they searched for |
| Precondition(s) | * The customer is logged into the web app * The customer has a list of cruises to select from in the web app |
| Assumption(s) | The customer wants to view more specific information about the cruise that is not displayed in the initial search results |
| Flow of Events | 1. Customer logs into the web app 2. Customer searches for their desired cruise to get more information about it 3. Customer clicks the name of the cruise 4. The web app displays specific information about the cruise: name, cruise line, departure date, arrival date, ports of call, price, map of itinerary, amenities, photos of the ship inside and out, types of cabins available, listing of stores and restaurants on the ship, ideal type of cruise for this cruise, links to videos about the cruise 5. Contact information for the booking agency and cruise line are displayed 6. Option to book from here is available if the customer has not booked this cruise |
| Postconditions | The full listing of amenities is displayed, the customer can view videos from this page, the customer can also jump to booking this cruise |

**Table 3**

*Booking a Cruise*

|  |  |
| --- | --- |
| Actor(s) | * Customer (Primary) * Cruise Web App (Supporting) * Booking Engine (Primary) * Customer’s Bank (Supporting) |
| Description of the Functionality | Main feature of the app: to book a cruise including finding the cruise, choosing it, making payment, entering passenger details, etc |
| Precondition(s) | * The customer is logged into the web app * The customer has found the cruise they wish to book * The customer is ready to pay either a deposit or for the full cruise |
| Assumption(s) | 1. The cruise still has cabins left 2. The customer has not already booked this cruise or another one which overlaps with this one’s voyage dates |
| Flow of Events | 1. Customer logs into the cruise web app 2. Customer finds their desired cruise through the search engine 3. Customer chooses the cruise to book 4. Web app displays the full details page is shown 5. Customer clicks the Book Cruise button 6. The web app displays a form for the customer to fill in passengers’ full contact details including birth dates, passport numbers, phone numbers, addresses. The customer clicks to advance to the next page. 7. The web app prompts the customer for amenities, cabin type, whether they need a flight to the departure location, and insurance options. The customer clicks to advance to the next page. 8. The web app prompts for payment details - if this is for a brand-new booking, the payment will be for the deposit and if this is for an existing booking, the payment will be for the remaining balance. 9. The web app forwards the payment information to the customer’s bank for processing and booking information to the booking engine. 10. Upon successful payment, the web app displays the full booking receipt on screen and forwards the same information to the customer’s e-mail. |
| Postconditions | The cruise is booked, and the customer and any involved other passengers have received their email with the receipt for the cruise. |

**Table 4**

*Modifying or Cancelling a Cruise*

|  |  |
| --- | --- |
| Actor(s) | * Customer (Primary) * Web App (Supporting) * Customer Services Representative (Supporting) * Booking Engine (Supporting) * Customer’s Bank (Supporting) |
| Description of the Functionality | Modify or cancel a cruise: modifications could be for adding airfare, pre/post-cruise hotel stays, adding or deleting a passenger, adding travel insurance, adding or deleting amenities, or cabin upgrades/downgrades, make sure the customer contacts customer service to verify cancellations |
| Precondition(s) | 1. The customer has an account on the web app 2. The customer has an existing cruise booking 3. Any deadlines the cruise line established for changes have not lapsed |
| Assumption(s) | The customer had a cruise booking that can be modified or cancelled |
| Flow of Events | 1. The customer logs into the cruise booking web app 2. Customer finds their cruise they wish to modify or cancel and chooses it 3. The customer is prompted to choose between modification or cancellation    1. If the customer chose to modify the cruise, the customer is prompted for the types of modifications they are allowed to make. For certain types of modifications, such as adding or deleting a passenger, the app will redirect the customer to contact customer service to verify exact details    2. If the customer chose to cancel the cruise, the app will redirect to call or video a customer services representative to verify the cancellation 4. The booking engine verifies modification or cancellation 5. The booking engine or customer services representative initiates a refund or upgrade charge to the customer’s bank 6. The web app sends the customer an email with the receipt showing confirmation details |
| Postconditions | The cruise was modified or cancelled successfully, and the customer got a refund or charged extra for upgrades, if applicable. The customer and any involved other passengers have received a receipt. |

**Table 5**

*Requesting room service or other amenities during a cruise*

|  |  |
| --- | --- |
| Actor(s) | * Customer/passengers (Primary) * Web app (Supporting) * Customer services representative (Supporting) * Person giving/performing the amenity (Primary) |
| Description of the Functionality | The customer requests for room service or other amenities offering on the cruise; they may also write a review on amenities they experienced. |
| Precondition(s) | The customer is on a cruise they booked |
| Assumption(s) | The customer is not already receiving said amenities |
| Flow of Events | 1. During the cruise, the customer wishes to request room service or another amenity 2. The customer opens the web app on their phone via the cruise ship’s WiFi signal to view the different amenities 3. The customer chooses their cruise in the app 4. The customer selects to view available amenities in the app 5. The amenity choices are shown for the customer to select 6. If necessary, the app prompts for a relative time and payment option 7. The customer submits their request from the web app 8. The web app sends the customer their receipt to their email 9. The web app displays where the customer needs to go to receive their amenities 10. After experiencing the amenities, the customer may write a review in the web app |
| Postconditions | 1. The customer has received their amenities and receipt. 2. [Optional] They may have also submitted a review for their amenities. |

**Table 6**

*Contact a customer services representative*

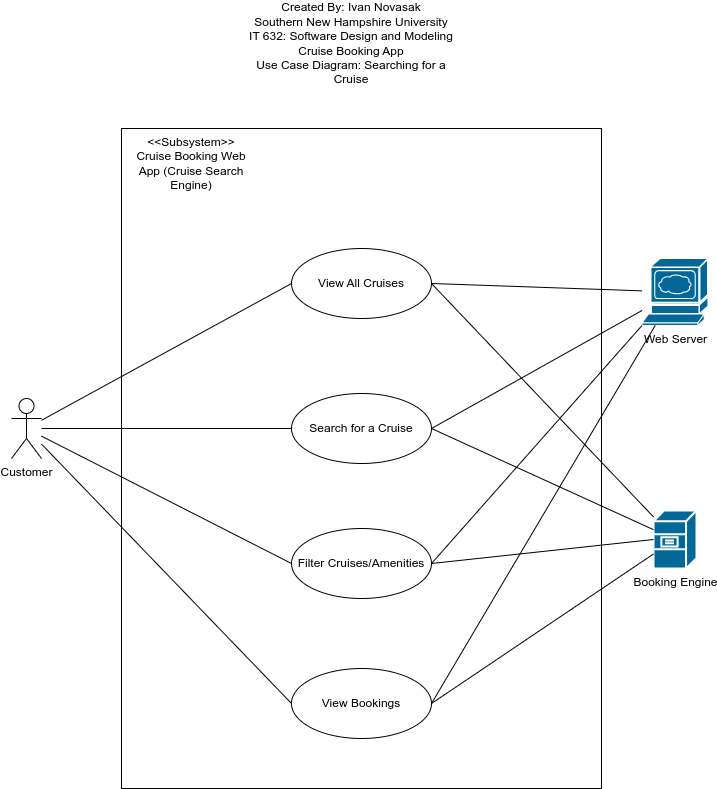
|  |  |
| --- | --- |
| Actor(s) | * Customer (Primary) * Web App (Supporting) * Customer Services Representative (Supporting) |
| Description of the Functionality | The customer should be able to contact a customer services representative by phone, video calls, or text chat within the web app |
| Precondition(s) | The customer is logged into the web app and needs help with a topic related to cruising or their bookings |
| Assumption(s) | The customer has decided they want to either call, video conference, or text chat with a representative |
| Flow of Events | 1. The customer logs into the web app 2. The customer clicks a button in the app that shows ‘Get Help’ 3. The app displays the options to the customer:    1. By phone    2. By video conference/Zoom    3. By text chat 4. The customer chooses their choice in the app 5. The phone or computer prompts to open the relevant app to contact customer service    1. If the customer chose ‘By Phone’, the app should just display the number embedded in a tel: link so the customer may either call from there directly if using their phone or display the number on screen if using a computer |
| Postconditions | 1. The customer has got their problem resolved by speaking with the representative. 2. The customer receives an optional survey for CSR performance. |

**UML Use Case Diagrams**

Figures 1–6 are the use case diagrams for each use case scenario, adapted from a cruise booking scenario by WSQ2DyeW, of EDrawMax (WSQ2DyeW, 2021).

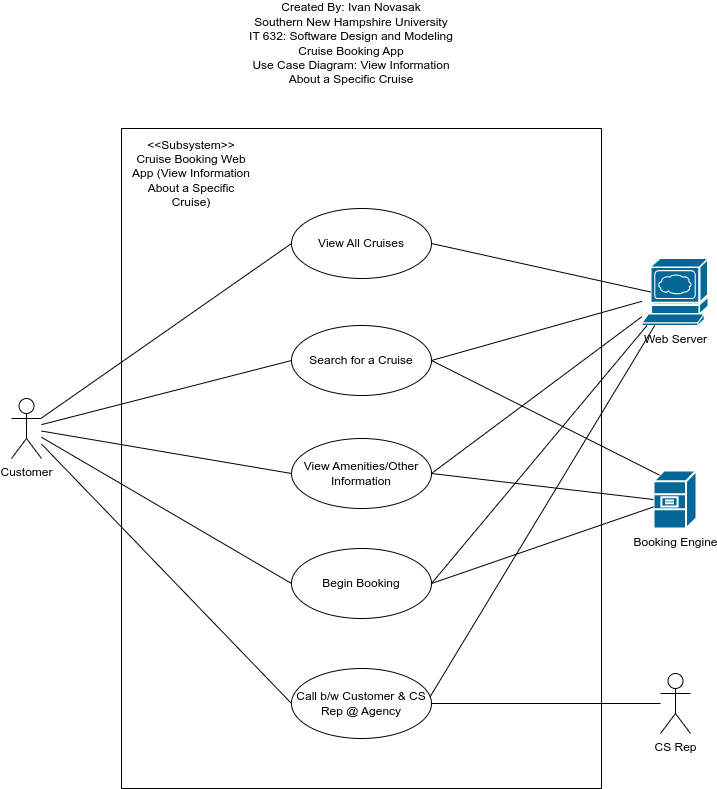
**Figure 1**

*Searching for a Cruise*



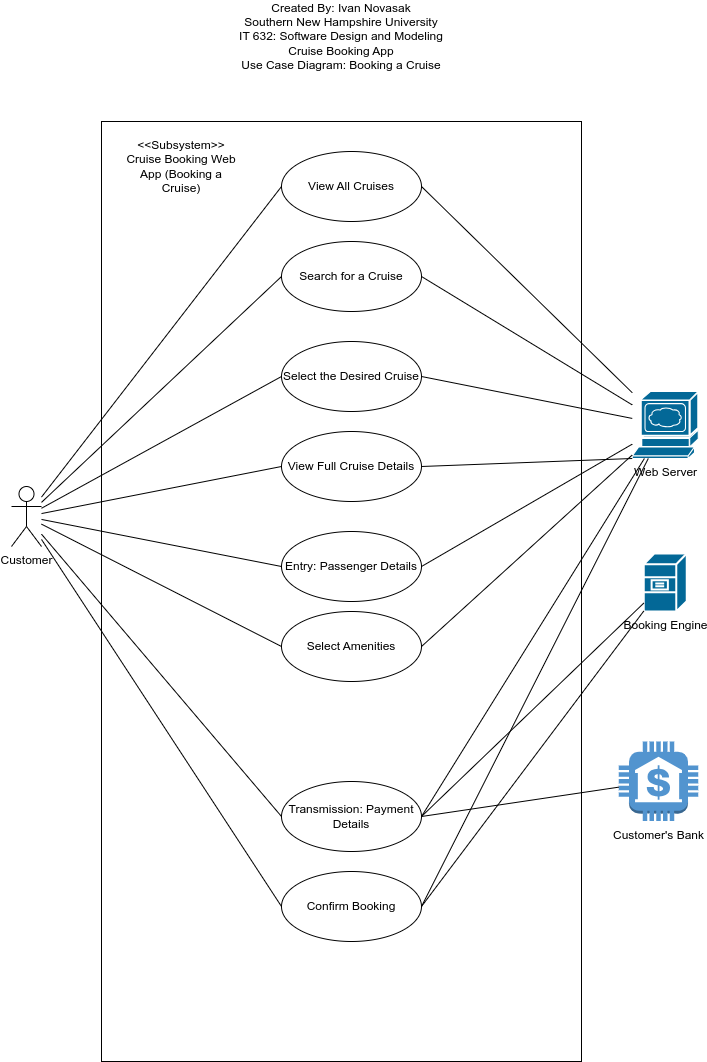
**Figure 2**

*View information about a specific cruise*

**

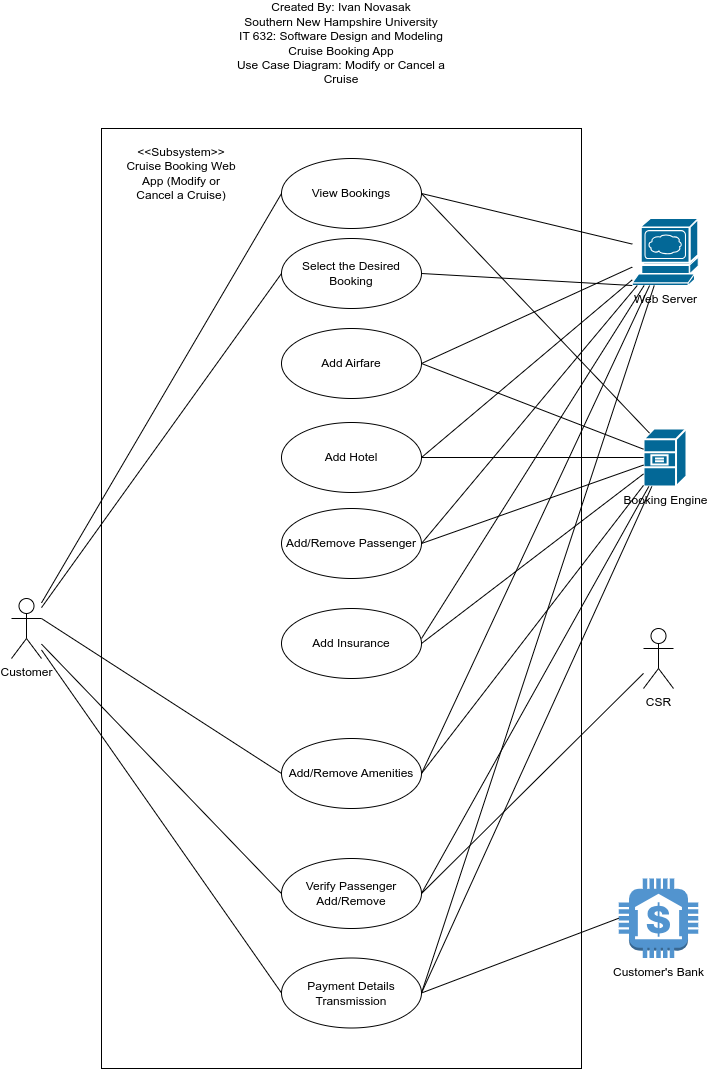
**Figure 3**

*Booking a cruise*

**

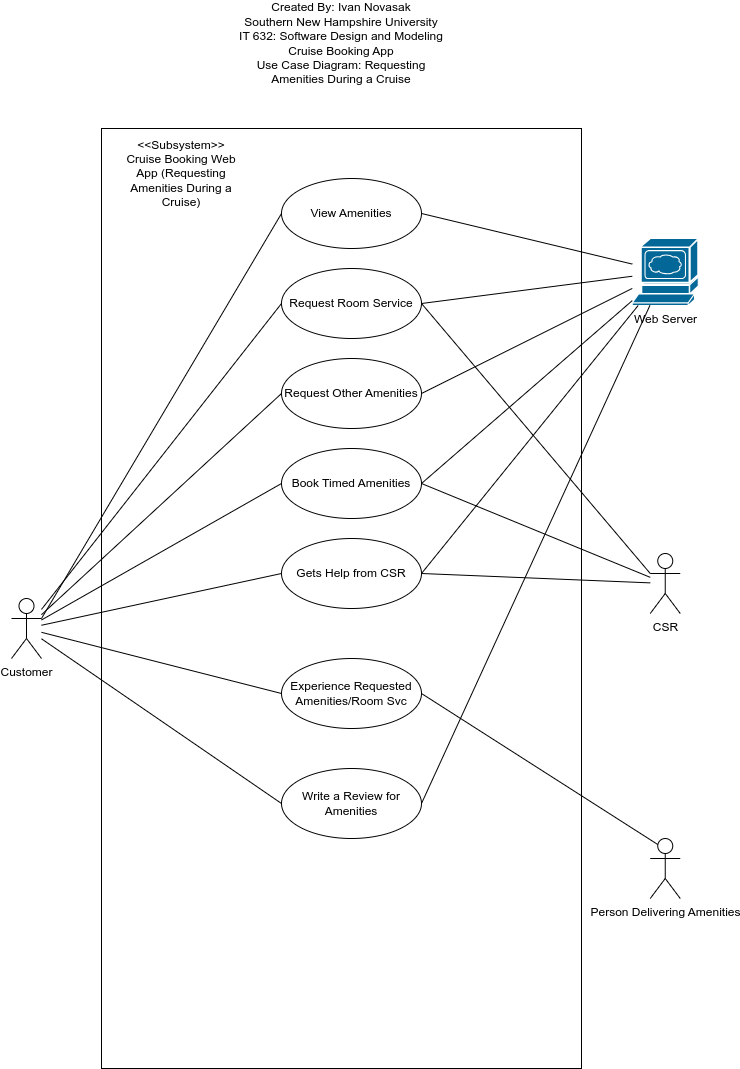
**Figure 4**

*Modify or cancel a cruise*

**

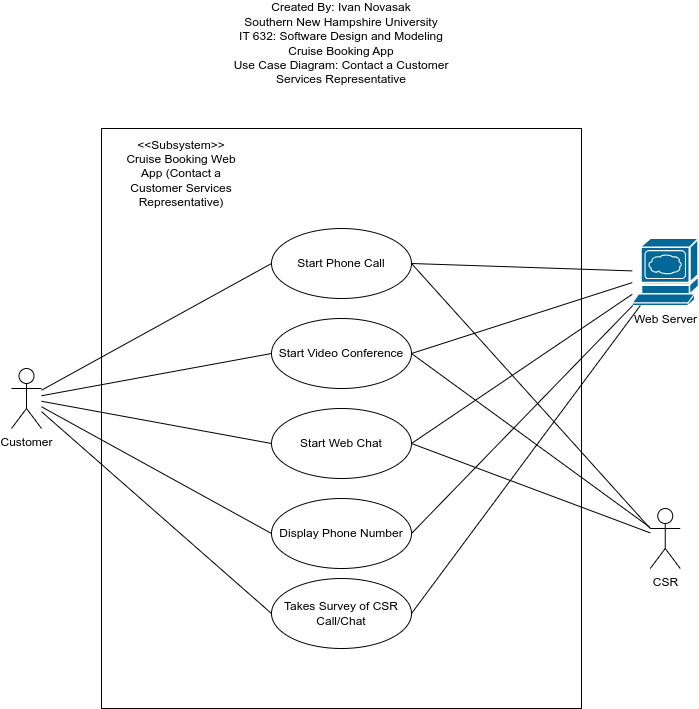
**Figure 5**

*Requesting amenities/room service during a cruise*

**

**Figure 6**

*Contact a customer services representative*



**Functional Requirements**

According to AltexSoft, functional requirements are requirements that describe what the software will do, features, the users’ requirements, user interface, and are defined by the users who will use the system (AltexSoft, 2023). They may also include authentication, authorisation, transaction handling, error logging, reporting, and backup/recovery (AltexSoft, 2023). For the cruise booking app, a listing of functional requirements based on the use cases can be as follows (Cruise Critic, 2022):

* Search engine for cruises
* View cruises
* View the customer’s cruise bookings
* Get help from a customer services representative by phone/video conferencing/text chat
* View cruise amenities
* Add or remove amenities for a cruise booking
* Pay for a cruise booking
* Add airfare, hotel, and insurance to a booking
* Email a receipt to the customer for a cruise booking or changes payment or refunds
* Filtering of cruises displayed based on cruise attributes:
  + Cruise lines
  + Origins
  + Destinations
  + Ports of call
  + Cabin type
  + Dates
  + Duration
  + Themes
  + Prices
  + Amenities
* Booking for multiple passengers
* Add amenities and request room service during the cruise using mobile phone and the cruise ship’s WiFi connection
* Web app is to be a central place the customer can search for, book, modify, pay for, and request amenities/room service during the cruise
* Customer can write reviews of the cruise, amenities, and customer services representative within the app
* Easy-to-use interface
* Videos and photos of cruise ship and available amenities in full details page about a specific cruise

**How will the use cases be used to design the architecture of this system (sequence and class diagrams)?**

The use case scenarios will be used to create sequence diagrams modelled after some hotel reservation diagrams made by Creately (Creately, n.d.). Namely the Flow of Events entries for each scenario will be used. The sequence diagrams provide a visualisation of the order in which different events take place, along with who is involved in each event (Larman, 2004, pp 176/177). Sequence diagrams help the developers visualise the timeline of when each event takes place, as well as what order they take place in and who is involved at each step. (Larman, 2004, pp 176/177).

According to GeeksforGeeks, UML class diagrams show a visualisation of objects relationships between them, and dependencies in software (GeeksforGeeks, 2024a). The class diagrams represent objects with rectangles, methods (software functions) by a smaller rectangle inside the object one, and attributes by another smaller rectangle inside the object rectangle (GeeksforGeeks, 2024a). An attribute or method may be public or private, which are designated by a plus or minus sign in front of each label, respectively (GeeksforGeeks, 2024a). Dependencies, inheritance, and relationships between objects are shown using arrows that link each outer rectangle with each other (GeeksforGeeks, 2024a). The class diagrams are a visual aid for developers who will be converting these diagrams into code, with adjustments as needed in the various iterations of the project (GeeksforGeeks, 2024a). For the cruise booking web app, an example of a function may be viewAllCruises. An example of an object may be Customer, with its attributes being the customer’s name, date of birth, phone number, address, passport number, etc. Another object may be Cruise, with its own relevant attributes like departureDate, price, and maxPassengers as some examples.

**Software Design**

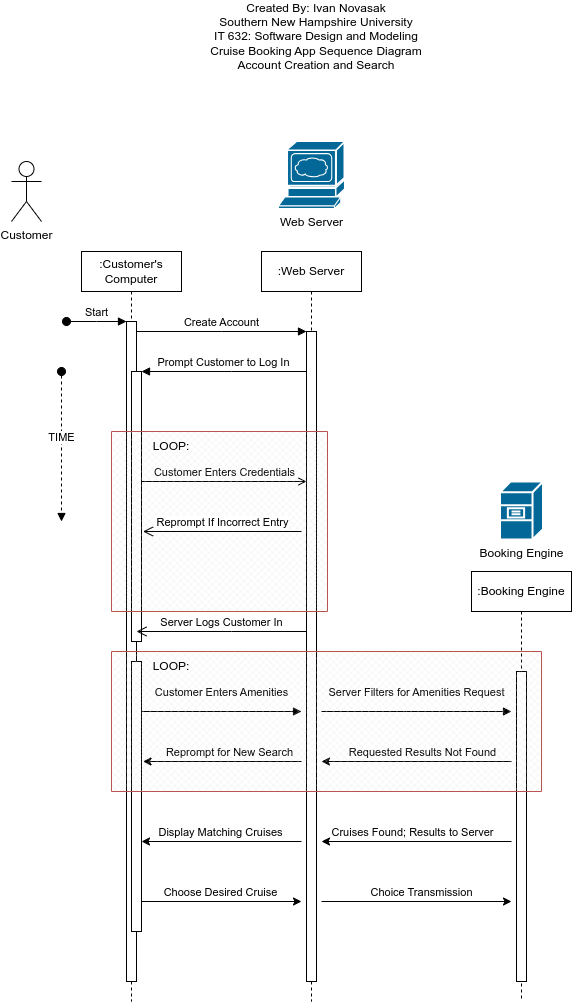
**UML Sequence Diagrams and Their Uses**

***Sequence Diagrams***

Figures 7 and 8 contain images of the sequence diagrams for searching and filtering for a cruise and booking a cruise once one has been selected. They are adapted from diagrams made by Amanda Athuraliya of Creately (Athuraliya, 2022). The objects and example images are placed at the top before the beginning of each object’s timeline. The arrows indicate the direction of each activity from the origin to the destination object. The boxes with the red outlines indicate loops for when the customer enters incorrect login or payment information. The box with a blue outline is for the extra activity of adding a flight and/or hotel to the cruise booking. It is labelled optional because not all passengers will need a flight and hotel at their cruise port city.

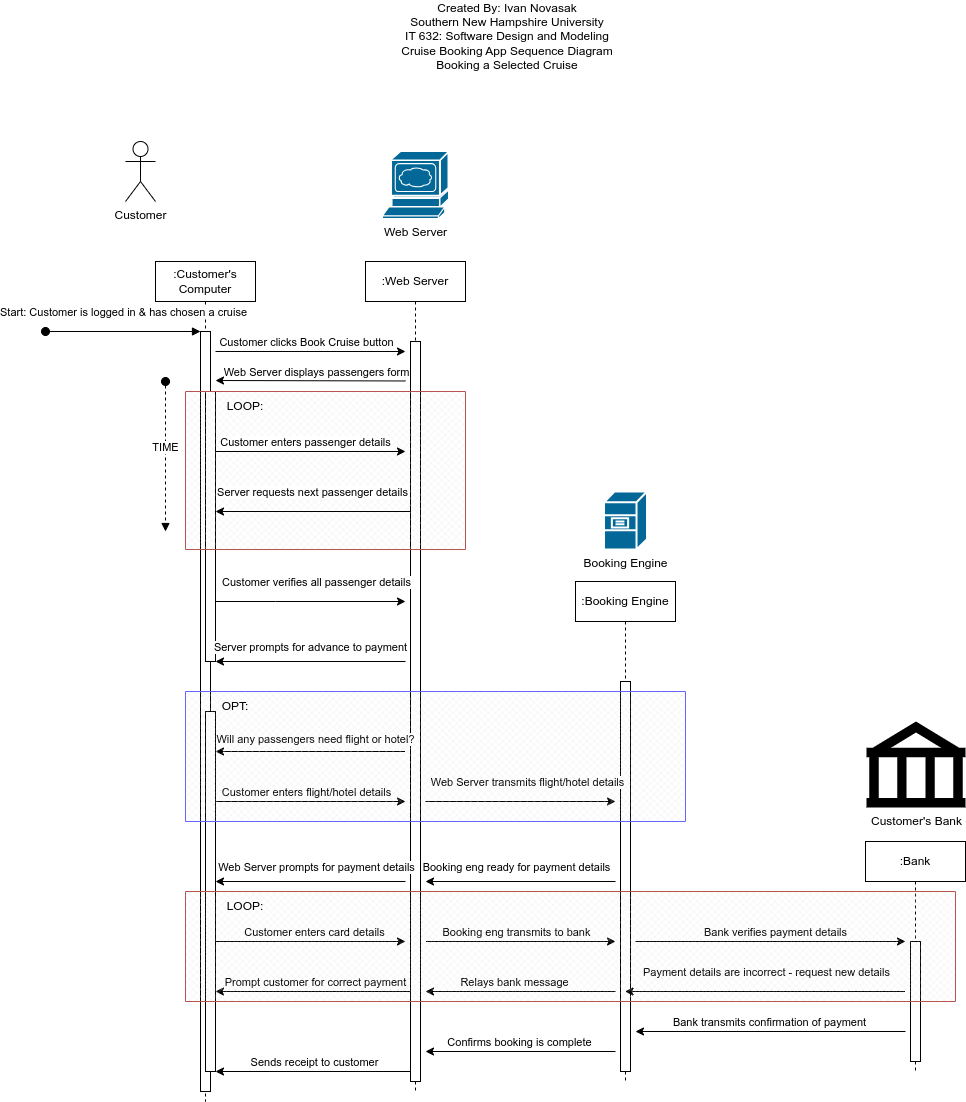
**Figure 7**

*Searching and filtering for a cruise based on amenities*



**Figure 8**

*Booking a selected cruise*



***Flow of Logic***

According to Lucidchart, the logical flow in a sequence diagram is carried by the arrows, which represent synchronous messages and ‘is used when a sender must wait for a response to a message before it continues’, requiring arrows for both the call and reply (Lucidchart, n.d.). The diagram is arranged with later time events being placed below earlier ones (Lucidchart, n.d.). Objects are represented by rectangles and skinny vertical rectangles, called activation boxes, are placed over the object lifelines representing activities in the timeline (Lucidchart, n.d.). Loops are used in both diagrams to represent both the user entering incorrect data in the forms and adding extra passengers and hotel rooms and/or flights.

***Object Interactions***

Both diagrams display the customer (and their computer), the web server, and booking engine as objects. The diagram for the booking process has the customer’s bank as an additional object due to the requirement for payment. The customer and the web server are present the entire time, whereas the booking engine and bank are present only in the later stages when the customer needs the cruise filtering results to appear, and when it is time for the payment to be processed.

***How Sequence Diagrams Fit in the Software Development Life Cycle***

According to Craig Larman in *Applying UML and patterns*, a sequence diagram is ‘is a picture that shows, for one scenario of a use case, the events that external actors generate, their order, and inter-system events’ (Larman, 2004, p 176). The diagram shows the involved actors, the actions they perform and await orders from other actors, messages/functions each actor must perform over time, and the timeline for the use case scenario. It also shows which actors are involved with which operation with arrows that point to blocks in the timeline that represent the functions and their corresponding actors. Not all actors are involved for all functions, and the diagram indicates this with the arrows and blocks. Loops could be shown in the diagram with the relevant functions being enclosed in a labelled, overlapping box that covers the arrows and names of the functions, which can be relevant for operations that may need to be repeated, such as if a person enters the wrong password or payment is rejected and is prompted for the correct or another one. The sequence diagram cannot be made until the use cases, actors, actions, and their scenarios are defined in order. It is always a good idea to have another involved person review the diagram to ensure the creator did not miss anything.

According to Brendon McConnell, sequence diagrams fit into the software development life cycle in the design stage, after use cases are defined but before formal classes are defined (McConnell, 2024). He says they ‘show how software objects and features interact in a particular sequence, helping designers and developers understand the dynamic behavior of the system they’re building’ (McConnell, 2024). These diagrams are alongside others, such as

use case diagrams, swimlane diagrams (used to visually chart who is involved in each portion of development), activity diagrams, flowcharts, deployment diagrams, and network diagrams (McConnell, 2024).

**UML Class Diagram**

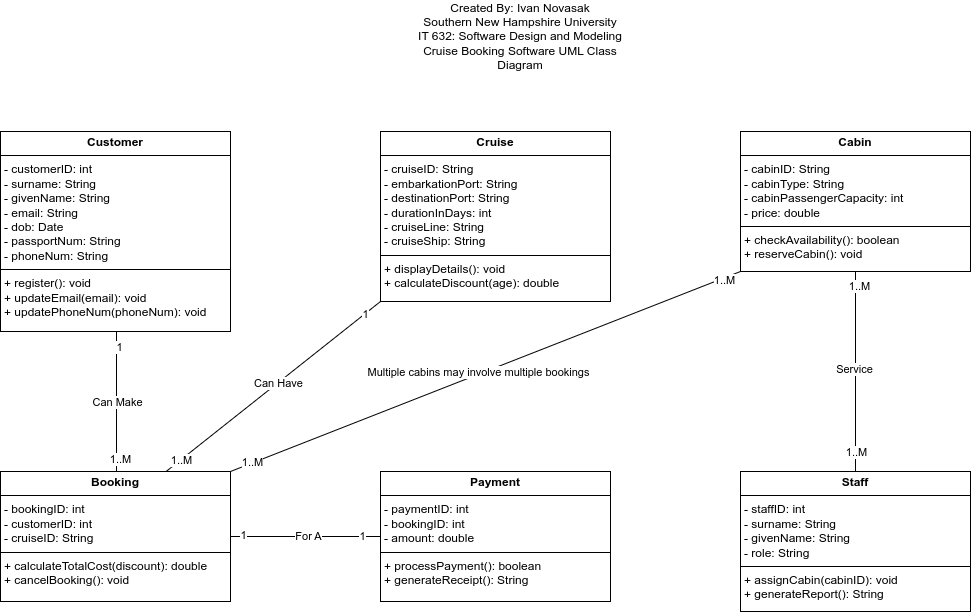
Figure 9 shows the UML class diagram for the cruise booking software. It is adapted from a UML class diagram design by Visual Paradigm on a retail order system (Visual Paradigm, n.d.). The big rectangles represent each object/class, with each rectangle being broken up into three smaller ones with the following components in each:

* The top section has the object/class’s name
* The central section has the class’s attribute/field names, their privacy status (minus sign for private, plus sign for public), and their data types
* The bottom section has the class’s method/function names, their privacy status (minus sign for private, plus sign for public), input parameter names, and return data types.

The relationships between these classes are shown by lines connected to the related classes and numbers near the end of the lines that come out of the rectangles. The number 1 refers to one instance of that class, and 1..M refers to many, but at least 1 (Larman, 2004, p 264; Visual Paradigm, n.d.).

**Figure 9**

*UML class diagram showing the objects and relationships between them*



***Structure of the Software System***

The software has the following relationships between its classes, shown in Table 7:

**Table 7**

*Relationships between classes*

|  |  |
| --- | --- |
| **Class Relations** | **Relationship** |
| Customer to Booking | One customer can make many bookings. (1-to-many) |
| Booking to Cruise | Many bookings are for one cruise / One cruise many have many bookings. (many-to-1) |
| Booking to Cabin | Many bookings can include one or more cabins. (many-to-many) |
| Payment to Booking | One payment is for one booking. (1-to-1) |
| Staff to Cabin | Many staff members can manage, assign, and service many cabins. (many-to-many) |

***How the UML Class Diagram Fits Within the Software Development Life Cycle***

According to GeeksforGeeks in the web page *Is UML part of SDLC?*, UML class diagrams are especially used in the design phase of a five-phase software development life cycle, but also in other phases, such as implementation, testing, and maintenance (GeeksforGeeks, 2024). Their full list of development phases is as follows (GeeksforGeeks, 2024):

1. Requirement Analysis
2. System Design
3. Implementation
4. Testing
5. Maintenance.

In the design and implementation phases, UML class diagrams are used to define the different classes, methods/functions, and attributes as a type of blueprint for coders to refer to (GeeksforGeeks, 2024). In the testing phase, multiple different UML diagrams are used to make test cases that will ensure the product will meet the end user’s requirements (GeeksforGeeks, 2024).

According to William Crawford and Jonathan Kaplan in their book *J2EE Design Patterns*, UML class diagrams are used in the high-level design process to ‘define the domain model for the application: specifically, the relationship of data objects within the system, the relationships between them, and the operations that they can perform or that can be performed on them’ (Crawford & Kaplan, n.d.). This is during the second phase of six total phases they list which run as follows (Crawford & Kaplan, n.d.):

1. Requirements-gathering
2. High-level design
3. Low-level design
4. Coding and unit testing
5. Integration testing
6. Deployment.

**Next Steps**

This final section is to describe how the cruise booking software can be implemented. According to Rahul Awati of Informa TechTarget, the software and associated code can be written iteratively with each iteration cycle having the following four phases (Awati, 2023):

1. Planning and analysis: This phase is used to define the software’s requirements that are in line with their business requirements and create any necessary database models and business logic. In the first iteration, this phase also contains a feasibility analysis and a project plan with schedule, budget, and tasks.
2. Design: In this phase, the software developers define the technical requirements and decide on which programming languages and frameworks will be used for the project. During the first iteration this is established, whereas in future iterations a continuation of the overall design with enhancements is performed.
3. Technical architecture and coding: This phase is where the development team builds versions of their software, with each iteration being a new version of the software to be tested.
4. Testing and maintenance: In this phase, developers check for bugs and any errors that are out of line with conforming to the business logic. Three types of testing are done: unit testing for individual code modules, integration testing to verify that the software is functioning correctly, and user acceptance testing to make sure the software performs to the users’ goals.

Each iteration is ‘timeboxed’, which means each one will last a specific defined amount of time with no changes in timing (Awati, 2023). The development team and management decide on the time units to use, but it typically is two to six weeks per iteration (Awati, 2023; Larman, 2004, pp 22/23). At the start of subsequent iterations, team members discuss both what went well and what needs to be improved upon, and the new objectives for the current iteration. Before the software is shipped to production, the team has the client fully review it to ensure it is performing to their full specifications and requirements (Awati, 2023).

**Possible Code for the Customer Class and Main Method**

The following Java code could be a possible way to do the cruise booking software’s Customer class and Main method as a first iteration of code that the development team could improve on in later iterations:

***The Customer Class***

public class Customer {

// Private attributes

private String customerID;

private String name;

private String email;

private String phoneNum;

// Default constructors

public Customer(String customerID, String name, String email, String phoneNum) {

this.customerID = customerID;

this.name = name;

this.email = email;

this.phoneNum = phoneNum;

}

public Customer() { // Default constructor

}

// Accessors and mutators (getters and setters)

public String getCustomerID() {

return customerID;

}

public void setCustomerID(String customerID) {

this.customerID = customerID;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public String getPhoneNum() {

return phoneNum;

}

public void setPhoneNum(String phoneNUm) {

this.phoneNum = phoneNum;

}

// User-defined (custom) Methods

public void register() {

System.out.println("Customer " + name + " has been registered.");

}

public void updateCustomerEmail(String newEmail) {

setEmail(newEmail);

System.out.println("Customer’s email address is updated to " + newEmail);

}

public void updateCustomerPhoneNum(String newPhoneNum) {

setPhoneNum(newPhoneNum);

System.out.println("Customer’s phone number is updated to " + newPhoneNum);

}

}

***The Main Method***

public class CruiseBookingSystem {

public static void main(String[] args) {

// Instantiate a new Customer object taking in a Customer ID, Name email address, and phone number

Customer = new Customer("CUST001", "Kaylyn Irizarry", "[irizarrykaylyn@example.ca](mailto:irizarrykaylyn@example.ca)", "+17865550113");

// Invoke the register method to register the customer

customer.register();

// Invoke a method to update Kaylyn’s email address to irizarrykay2@example.ca

customer.updateContactInfo("irizarrykay2@example.ca");

// Instantiate a new Cruise object taking in a cruise ID, destination, and number of days

Cruise = new Cruise("CRUISE001", "Dominican Republic", 7);

cruise.displayDetails();

// Instantiate a new Cabin object taking in a Cabin ID, Cabin Type, and price

Cabin = new Cabin("CAB001", "Luxury", 2000);

// Invoke a method to reserve the cabin

cabin.reserveCabin();

// Instantiate a new Booking object taking in a Booking ID, Customer ID, and Cruise ID

Booking booking = new Booking("BOOK001", "CUST001", "CRUISE001");

// Invoke a calculation method that takes in the cost and discount ratio

booking.calculateTotalCost(2000, 0.1);

// Instantiate a new Payment object taking in a Payment ID, Booking ID, and amount [ the amount was calculated by cabin cost - (discount rate\*cabin cost) ]

Payment payment = new Payment("PAY001", "BOOK001", 1800);

// Invoke the method to process the payment

payment.processPayment();

}

}

# References

AltexSoft. (2023, November 30). *Functional and nonfunctional requirements: Specification and types*. AltexSoft. <https://www.altexsoft.com/blog/functional-and-non-functional-requirements-specification-and-types/>

Athuraliya, A. (2022, December 12). *Sequence Diagram Tutorial – Complete Guide with Examples*. Creately. <https://creately.com/guides/sequence-diagram-tutorial/>

Crawford, W., & Kaplan, J. (n.d.). *2.3 UML and Software Development Lifecycles - J2EE Design Patterns [Book]*. O’Reilly Online Learning. <https://www.oreilly.com/library/view/j2ee-design-patterns/0596004273/ch02s03.html>

Creately. (n.d.). *hotel booking system sequence diagram.* Creately. <https://creately.com/diagram/example/inhrk7dm1/hotel-booking-system-sequence-diagram>

Cruise Critic. (2022, October 12). *How to book a cruise: Booking a Cruise FAQ - Cruise Critic*. The Independent Traveler, Inc. <https://www.cruisecritic.com/articles/how-to-book-a-cruise-booking-a-cruise-faq>

Cunard. (2024). *7 nights – Westbound Transatlantic Crossing – June 2025 – Cunard*. Cunard Line. <https://www.cunard.com/en-us/find-a-cruise/M517/M517>

Desai, J. (2023, December 27). *Web application vs desktop application: pros and cons*. Positiwise Software Private Limited. <https://positiwise.com/blog/web-application-vs-desktop-application-pros-and-cons>

GeeksforGeeks. (2024a, October 18). *Class Diagram | Unified Modeling Language (UML)*. GeeksforGeeks. <https://www.geeksforgeeks.org/unified-modeling-language-uml-class-diagrams/>

GeeksforGeeks. (2024b, February 23). *Is UML part of SDLC?* GeeksforGeeks. <https://www.geeksforgeeks.org/is-uml-part-of-sdlc/>

Larman, C. (2004). *Applying UML and patterns: An Introduction to Object-oriented Analysis and Design and Iterative Development*, Third Edition. Pearson.

Lucidchart. (n.d.). *UML Sequence Diagram Tutorial.* Lucid Software Inc. <https://www.lucidchart.com/pages/uml-sequence-diagram>

McConnell, B. (2024, August 7). *How to use visual diagramming to unlock better SDLC collaboration*. Miro. <https://miro.com/blog/sdlc-stages/>

Awati, R. (2023, June 13). *iterative development*. Informa TechTarget. <https://www.techtarget.com/searchsoftwarequality/definition/iterative-development>

Visual Paradigm. (n.d.). *UML Class Diagram Tutorial*. Visual Paradigm. <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/>

WSQ2DyeW. (2021, October 27). *Use case Diagram for cruise online booking system | EDrawMax Templates*. EDrawMax. <https://www.edrawmax.com/templates/1009782/>