

Name: K.C. Berenger

Student Reference Number: 10673083

|  |   |
|--|---|
| Module Code: PUSL3106  | Module Name: Design Patterns and Software Engineering                 |
| Coursework Title: Web based Hotel Booking System for Hotel Green View(Design Patterns and Software Engineering Coursework) |   |
| Deadline Date: 30 April 2021   | Member of staff responsible for coursework: Prof. Prasad M. Jayaweera |

Programme: BSc (Honours) Software Engineering

Please note that University Academic Regulations are available under Rules and Regulations on the University website [www.plymouth.ac.uk/studenthandbook](http://www.plymouth.ac.uk/studenthandbook)

Group work: please list all names of all participants formally associated with this work and state whether the work was undertaken alone or as part of a team. Please note you may be required to identify individual responsibility for component parts.

10673076 - P.G.G.M.B. Bandara  
 10673083 - K.C. Berenger  
 10673116 - I.A. Dissanayake  
 10673118 - M.D. Dissanayake  
 10673123 - S.S.D.H. Fernando  
 10673260 - M.A.O.G. Kithmina

***We confirm that we have read and understood the Plymouth University regulations relating to Assessment Offences and that we are aware of the possible penalties for any breach of these regulations. We confirm that this is the independent work of the group.***

Signed on behalf of the group: K.C. Berenger

Individual assignment: ***I confirm that I have read and understood the Plymouth University regulations relating to Assessment Offences and that I am aware of the possible penalties for any breach of these regulations. I confirm that this is my own independent work.***

Signed: \_\_\_\_\_

Use of translation software: failure to declare that translation software or a similar writing aid has been used will be treated as an assessment offence.

I \*have used/not used translation software.

If used, please state name of software.....

Overall mark \_\_\_\_\_%      Assessors Initials \_\_\_\_\_      Date \_\_\_\_\_

# CONTEXT

|  |          |
|--|----------|
| 1. Introduction.....   | 04       |
| 1.1. Acknowledgement.....  | 04       |
| 1.2. Abstract.....   | 04       |
| 1.3. Introduction to the Coursework.....                         | 05       |
| 2. Objectives.....   | 05       |
| 3. Scope.....  | 06       |
| 3.1. Hotel staff of Green View.....                              | 06       |
| 3.2. Guests.....   | 06       |
| 3.3. System.....   | 06       |
| 4. Obstacles faced with a manual Hotel Booking System.....       | 07       |
| 5. Why use a web-based System?.....                              | 07       |
| 6. Potential benefits from the proposed System.....              | 07       |
| 7. System Requirement Development.....                           | 08       |
| 7.1. Assumptions.....  | 08       |
| 7.2. Use Case Diagram.....                                       | 09       |
| 7.3. Use Case Descriptions.....                                  | 10       |
| 7.3.1. Use Case Login.....                                       | 10       |
| 7.3.2. Use Case Registering Details.....                         | 11       |
| 7.3.3. Use Case Update Room Details.....                         | 12       |
| 7.3.4. Use Case View Room Details.....                           | 13       |
| 7.3.5. Use Case Search and View available Rooms.....             | 14       |
| 7.3.6. Use Case Book available Rooms.....                        | 15       |
| 7.3.7. Use Case Register Payment.....                            | 16       |
| 7.3.8. Use Case View Booking.....                                | 17       |
| 7.3.9. Use Case Update Room Availability.....                    | 18       |
| 7.3.10. Use Case Update Guest Details on Check out or Fallback.. | 19       |
| 7.3.11. Use Case Generate Daily Income Report.....               | 20       |
| 7.3.12. Use Case Cancel Booking.....                             | 21       |
| 7.3.13. Use Case Logout.....                                     | 22       |
| 8. System Analysis and Domain Modelling.....                     | 23       |
| 8.1. Functional View.....  | 23       |
| 8.1.1. Activity Diagram.....                                     | 23       |
| 8.1.2. State Machine Diagram.....                                | 24       |
| 8.2. Static View.....  | 25       |
| 8.2.1. Class Diagram.....  | 25       |
| <b>Code generation with ArgoUML screenshots as evidence</b>      | 26       |
| 8.3. Dynamic View.....   | 27       |
| 8.3.1. Collaboration Diagram.....                                | 27,28,29 |
| 8.3.2. Sequence Diagram.....                                     | 30,31,32 |
| 9. System Design.....  | 33       |
| 9.1. Methodology.....  | 33       |
| 9.2. Analysis of Traditional Hotel Booking System.....           | 33       |
| 9.3. Design for Proposed System.....                             | 33       |

|        |   |    |
|--------|---|----|
| 9.4.   | Users and their Characteristics.....            | 34 |
| 9.4.1. | System / Super admin.....                       | 34 |
| 9.4.2. | Hotel administration (Hotel Staff).....         | 34 |
| 9.4.3. | Guests (Guest).....                             | 34 |
| 9.5.   | Workflow of the Proposed System.....            | 35 |
| 9.5.1. | Login.....                                      | 35 |
| 9.5.2. | Bookings.....                                   | 35 |
| 9.5.3. | Rooms (Function, Single BR, and Double BR)..... | 35 |
| 9.5.4. | Reports generated.....                          | 35 |
| 9.6.   | Implementation Requirements.....                | 36 |
| 9.7.   | System Requirements.....                        | 36 |
| 9.7.1. | Hardware Requirements.....                      | 36 |
| 9.7.2. | Software requirements.....                      | 36 |
| 10.    | Conclusion.....                                 | 37 |
| 10.1.  | Individual Contribution.....                    | 37 |
| 10.2.  | References.....                                 | 38 |

# 1. Introduction

## 1.1 Acknowledgement

This is to honor and show appreciation to the partakers of the completion of this project. We thank, our in charge leader for the module, Design patterns and Software Engineering, Prof. Prasad Jayaweera for his guidance and support. To the team, 'Give yourself a pat on the back, for a job well done'!

## 1.2 Abstract

The given project is expected to suffice software development by adopting professional Software Engineering practices, to cover typical real-world systems.

The scenario follows a Hotel, Green View which incorporates function rooms, single and double bedrooms for lodging. A real time transaction system is required to book, administrate and generate reports based on income. The deliverable requires us to build up a requirement specification based on a through feasibility study, a System Analysis, Domain Modeling and System Design that supports and incorporates agile development while meeting time constraints given.

Module Name: Design Patterns and Software Engineering

Module Code: PUSL3106

Course work title: Web Based Hotel Management system for Hotel Green View

Group title: **Team 3**

Deadline: 30<sup>th</sup> April 2021

### **1.3 Introduction to the coursework**

Guarantying information at a real time pace, to users of the Internet Database Driven Websites are the majority in the field of IT and administration. Its dynamic nature and embedded Human Computer Interaction (HCI) Principals, makes it easy for Internet users to occupy in transactions. The reduced chances for errors, high scalability, and ease of use make Database driven websites popular in most business sectors. As such, our project is a system of similar caliber.

A hotel management system, for Hotel Green View consists of two main stakeholders.

1. Hotel administration
2. Guest (Guest)

In this system, both guest and Hotel Administration can login at free will and carry out relevant dealings. The hotel administration would carry out tasks such as registering room details based on category, updating Guest details and confirming bookings while a Guest can search and book for the available rooms under the three categories,

1. Function rooms at price 200 000/= LKR
2. Single bed rooms at price 20 000/= LKR
3. Double bed rooms at price 40 000/- LKR

At the end of the day the system would generate a daily income report under above three categories to calculate and predict profit.

## **2. Objectives**

The main goal of this project is to analyze and design an immersive web-based management system that would search availability of free rooms and display for Guest booking while generating reports monetarily statistics. This would be a fully automated system that would overtake the labor-intensive work of a manual process.

### 3. Scope

The to be designed system can be shaped when categorizing the requirement specification under a work break down structure (WBS). We will look at the main business case and objectives according to the actors partaking. (Edwards, 2020)

#### 3.1 Hotel Staff of Green View

While being able to login according to a specific ID registered by the super Admin, the Hotel administration would have an overview of reports,

1. Available rooms for booking – functions
2. Available rooms for booking – single BR
3. Available rooms for booking – DBR
4. The current days income

displayed on the Dashboard. They will also have a view of the details of guests already booked in, and guests who have reserved for a booking in. The administration will also have a feature to perform operations on room details (E.g., price changes on days of seasonal changes) when a guest wishes to leave or check out a head of the time given, the staff can manually intercede and edit necessary details. At the end of the day, when the system generates a report on income, the administration can view and calculate dealings after.

#### 3.2 Guests

A Guest can login at free will and check up on available rooms for booking under the three categories. He/ She can book a room preferred and enter relevant details and proceed to payment either offline or online. A Guest can cancel a booking under any circumstance given. He / She has not checked in.

#### 3.3 System

Would enable a daily income report to be generated by rounding up all information on availability, check ins and outs from all three categories of rooms. When a checkout reaches its last hour, the system will automatically register checkout and availability for booking again. Once booked, the system generates a mail to be sent to the guest. In it, a confirmation of booking and room number allocated in randomly picked through the filtering process.

#### **4. Obstacles faced with a manual hotel booking system**

1. When booking, a client would have to call or head to the location then and there. The same goes with the payment process as well.
2. When registering, details of Guests and updating the lodgers upon checking in and out data can be misplaced or lost.
3. When checking availability of rooms under 3 categories, it is hard to handle the volume of data.
4. When trying to cancel a booking, a client would again have to follow the same process under stage 1.
5. When generating reports manually, the statistics can be inaccurate, since all data necessary for calculation can be invalid as well.

#### **5. Why use a Web based system?**

1. Less labor intensive
2. Cuts down on cost
3. Saves time
4. Competitive advantage over external forces.
5. Guest and staff convenience
6. Secure and more reliable

#### **6. Potential benefits from proposed system**

1. Increases revenue
2. Can add on multiple bookings
3. 24 / 7 operations
4. Cuts work for hotel staff
5. Receptionist do not need to handle important data such as Guest details over a phone as such.

## 7. System requirement development

With the purpose of code generation, we have used a free and open-source UML (Unified Modelling Language) Diagramming Application named ArgoUML, in which the platform is supported by Java SE, for designing of Class Diagram and such. We have also used Draw.io which is a free and open-source Use Case Tool for the designing of Sequence and Collaboration Diagrams and such. (Walker, 2005) (Meier, 2017)

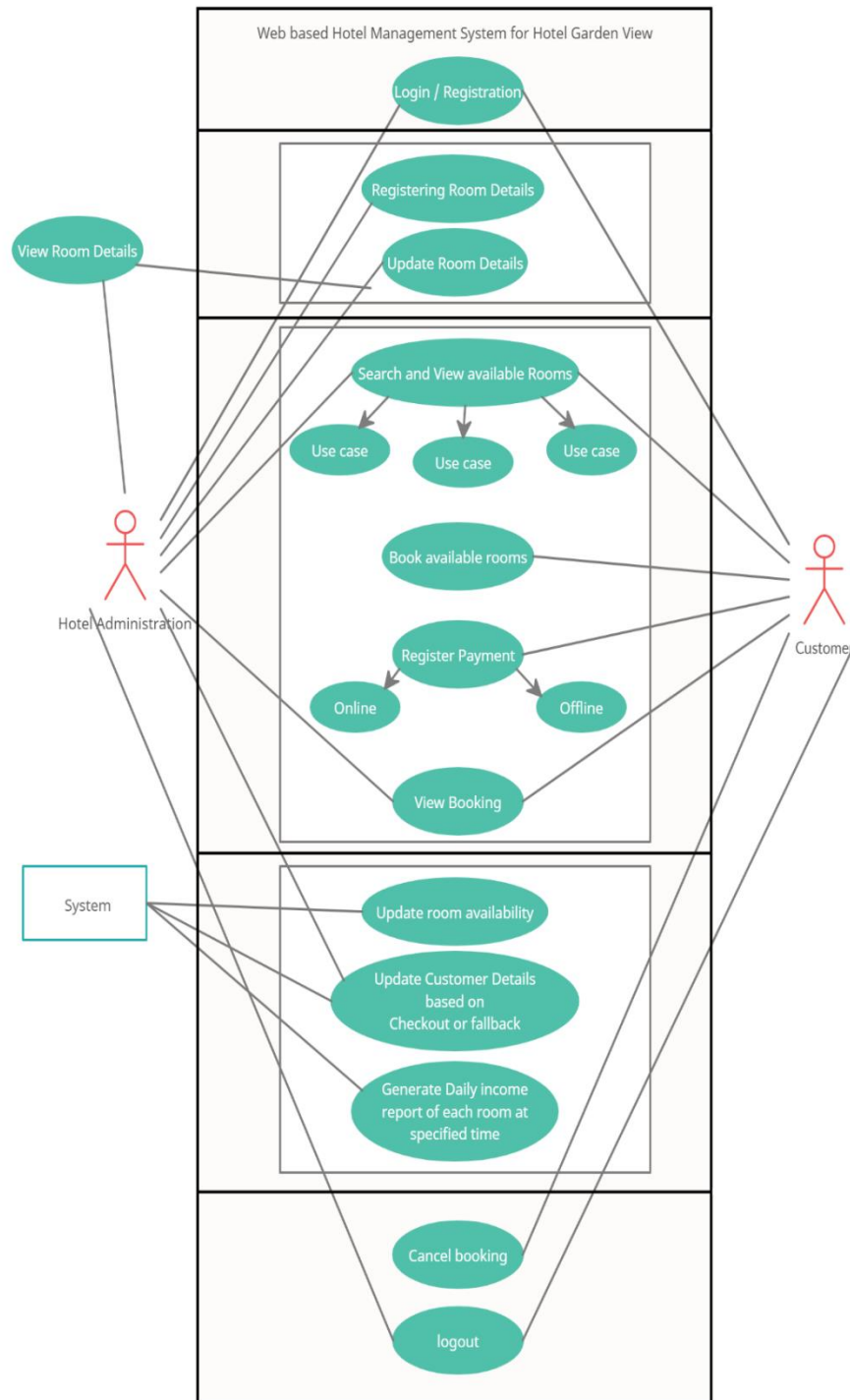
### 7.1 Assumptions

1. The registration of Hotel Staff happens through a network built by a super admin.
2. The registration of Guests occurs when a booking is made as Guest details are necessary for the process.
3. Hotel staff can manually update room availability just like the system auto loads upon checking in and checking out, if unseen circumstances take place.
4. Once a booking takes place, according to the filtered choice, a random selection of one room number from the available rooms will be done.



## 7.2 Use Case Diagram

Use Case Diagrams emphasize networks and connections between its actors and attributes. The relationships and priority can be graft into an easy to understand presentational view. (Larson, 2004) (Kupe Kupersmith)



### 7.3. Use Case Descriptions

#### 7.3.1. Use Case Login

|                                  |  |   |
|----------------------------------|--|---|
| Use Case No.                     | Uc - 01  |   |
| Use Case Name                    | Login  |   |
| Priority                         | High   |   |
| Actor                            | Hotel administration, Guest  |   |
| Description                      | If this use case is successful the administration at Green View and the Guest would be able to login to the system   |   |
| Pre-condition                    | None   |   |
| Post condition                   | Login success  |   |
| The fundamental course of action | User action  | System responses  |
|                                  | 1. Actor is on the Home page for login<br><br>3. Guest/Hotel administration enters user credentials and clicks the login button  | 2. System requests for login credentials from Guest/ hotel administration<br><br>4. System runs verification on credentials<br><br>5. Hotel administration/Guest successfully logs in<br><br>6. Use case exit |
| An alternative course of action  | 3.1. If user credentials are incorrect or not valid, the program will pop a message for validation and would allow re-filling of credentials, resuming back to stage 3 |   |

### 7.3.2. Use Case Registering Details

|                                  |   |   |
|----------------------------------|---|---|
| Use Case No.                     | Uc - 02   |   |
| Use Case Name                    | Registering room details  |   |
| Priority                         | High  |   |
| Actor                            | Hotel administration  |   |
| Description                      | This use case allows the hotel administration to enter details of all the rooms at Hotel Green View according to categories function rooms, single bed rooms, double bed rooms. |   |
| Pre-condition                    | Uc – 01   |   |
| Post condition                   | Registration of room details successful   |   |
| The fundamental course of action | User action   | System response   |
|                                  | 1. Administration needs to enter details of rooms<br><br>2. Fill in (new) room details form page loads<br><br>5. Hotel administration enters data and clicks submit             | 3. System displays room details form to be filled<br><br>4. System requests details such as room no, room type, price, room size, view to be entered<br><br>6. System confirms and validates data<br><br>7. System displays a successful save message<br><br>8. Use case exit |
| An alternative course of action  | 5.1. If the fields given in the form are empty, the system will show a message asking for fields to be filled while re-directing to stage 4                                     |   |

### 7.3.3. Use Case Update Room Details

|                                  |  |   |
|----------------------------------|--|---|
| Use Case No.                     | Uc - 03  |   |
| Use Case Name                    | Update room details  |   |
| Priority                         | High   |   |
| Actor                            | Hotel administration   |   |
| Description                      | This use case allows the hotel administration to update/edit details of rooms divided into categories function rooms, single bed rooms, double bed rooms   |   |
| Pre-condition                    | Uc – 01, Uc – 04, Uc – 05  |   |
| Post condition                   | Updating details of rooms registered   |   |
| The fundamental course of action | User action  | System responses  |
|                                  | 1. Administration needs to update details of rooms registered<br><br>2. Update room details in the room details form page<br><br>5. Hotel administration enters room no. and clicks view<br><br>9. User enters new and correct data and clicks the update button | 3. System displays room details form to be filled<br><br>4. System requests for room no to be entered to gather current data on specified room<br><br>6. System confirms and validates room no<br><br>7. System displays data already saved under room no<br><br>8. System prompts user to fill in new correct data to the relevant fields<br><br>10. System confirms and validates the data<br><br>11. System displays a successfully updated message<br><br>12. Use case exit |
| An alternative course of action  | 5.1. If entered room no is invalid a message displaying validation would appear routing back to stage 4<br><br>9.1. If the correct updates aren't entered and option to update routing back to stage 4 is optional   |   |

#### 7.3.4. Use Case View Room Details

|                                  |   |  |
|----------------------------------|---|--|
| Use Case No.                     | Uc - 04   |  |
| Use Case Name                    | View room details   |  |
| Priority                         | Medium  |  |
| Actor                            | Hotel administration  |  |
| Description                      | This use case allows the administration to view the room details registered or updated                                |  |
| Pre-condition                    | Uc – 02, Uc – 03, Uc – 05   |  |
| Post condition                   | The details saved or updated are displayed successfully   |  |
| The fundamental course of action | User action   | System responses   |
|                                  | 1. Hotel administration needs to view room details entered<br><br>2. A new page that retrieves room details is loaded | 3. System displays room details in grid format<br><br>4. System displays a message to confirm 'successfully retrieved'<br><br>5. Use case exit |
| An alternative course of action  | The system moves to phase 5 after data retrieval  |  |

### 7.3.5. Use Case Search and View available Rooms

|                                  |   |  |
|----------------------------------|---|--|
| Use Case No.                     | Uc - 05   |  |
| Use Case Name                    | Search and view available rooms   |  |
| Priority                         | High  |  |
| Actor                            | Guest, hotel administration   |  |
| Description                      | This use case allows both Guest and hotel administration to search up rooms that are available for booking  |  |
| Pre-condition                    | Uc – 01, Uc – 10, Uc – 11, Uc – 13  |  |
| Post condition                   | The details of rooms free for booking are successfully displayed  |  |
| The fundamental course of action | User action   | System responses   |
|                                  | 1. Actors are on the page that enable a search option<br><br>4. Actors select type of room accordingly (Eg: function, single bed room, Double bed room) and click the filter button | 2. System displays available rooms<br><br>3. System requests filtering on type of room<br><br>5. System confirms and validates data<br><br>6. System displays rooms available for booking according to filtering<br><br>7. Use case exit |
| An alternative course of action  | 4.1. Once the selected type of room is selected and there is no availability due to all being booked, the system would move back to stage 4   |  |

### 7.3.6. Use Case Book available Rooms

|                                  |  |   |
|----------------------------------|--|---|
| Use Case No.                     | Uc - 06  |   |
| Use Case Name                    | Book available rooms   |   |
| Priority                         | High   |   |
| Actor                            | Guest  |   |
| Description                      | This use case allows the Guest to book a room of any type based on its availability  |   |
| Pre-condition                    | Uc - 05  |   |
| Post condition                   | A booking of selected room is successfully completed   |   |
| The fundamental course of action | User action  | System responses  |
|                                  | 1. Guest is on the selected room panel<br><br>2. Guest clicks button booking and gets redirected to a registration form<br><br>5. User fills in the form with relevant data and clicks the book now button | 3. System displays registration form to be filled<br><br>4. System requests fields such as name, NIC, email, contact, no of days to be filled in<br><br>6. System validates and verifies data<br><br>7. System displays a select payment option<br><br>8. Use case exit |
| An alternative course of action  | At 5.1. if relevant required fields are not filled in, the system would display an error message prompting for all fields to be filled while reverting to stage 4  |   |

### 7.3.7. Use Case Register Payment

|                                  |  |   |
|----------------------------------|--|---|
| Use Case No.                     | Uc - 07  |   |
| Use Case Name                    | Register payment   |   |
| Priority                         | High   |   |
| Actor                            | Guest  |   |
| Description                      | This use case allows Guest to select a payment option for the booking made   |   |
| Pre-condition                    | Uc – 06  |   |
| Post condition                   | A payment method selected.<br>Based on the selection, either you can pay on visit or if payment is done online, user will be directed to a payment portal of choosing                  |   |
| The fundamental course of action | User action  | System responses  |
|                                  | 1. Guest wishes to select payment method to complete booking transaction<br><br>2. Guest re-directed to a selection of offline or online payment<br><br>5. User selects payment option | 3. System displays options for payment.<br><br>4. System prompts user to select payment option<br><br>6. Selection confirmed<br><br>7. Booking is confirmed based on selection process completion<br><br>8. System displays a booking confirmed message and generates a customized email automatically sent<br><br>9. Use case exit |
| An alternative course of action  | If in 5.1. payment option not selected system prompts for it by routing to stage 4   |   |



### 7.3.8. Use Case View Booking

|                                  |  |  |
|----------------------------------|--|--|
| Use Case No.                     | Uc – 08  |  |
| Use Case Name                    | View booking   |  |
| Priority                         | Medium   |  |
| Actor                            | Guest, Hotel administration  |  |
| Description                      | The actor can choose to view booking made and all details in relevance to the bookings   |  |
| Pre-condition                    | Uc – 01, Uc – 07, Uc – 12  |  |
| Post condition                   | A list of all bookings made is displayed in format   |  |
| The fundamental course of action | User action  | System responses   |
|                                  | <p>1. Guest/Hotel administration is on view booking page<br/>[Based on actors, Guest would see his/her bookings, while administration would be able to see all bookings]</p> <p>2.2. In case of administration, hotel staff fills in booking ID or just views all bookings</p> | <p>2. System displays relevant bookings according to actors</p> <p>2.1. In case of administration, he/she can view bookings according to booking ID automatically generated or all bookings</p> <p>3. Bookings are displayed</p> <p>4. System displays confirmation of retrieval</p> <p>5. Use case exit</p> |
| An alternative course of action  | 2.2.1. In case of hotel staff filling booking ID, if it is invalid the system would validate and pop an error while re-directing to stage 2.2.   |  |

### 7.3.9. Use Case Update Room Availability

|                                  |   |   |
|----------------------------------|---|---|
| Use Case No.                     | Uc – 09   |   |
| Use Case Name                    | Update room availability  |   |
| Priority                         | High  |   |
| Actor                            | System  |   |
| Description                      | Once a booking is made, changed or cancelled, the system would automatically update room availability under the three categories for future reservation |   |
| Pre-condition                    | Uc – 06, Uc – 10, Uc – 12   |   |
| Post condition                   | A list of available rooms is displayed for bookings and viewing   |   |
| The fundamental course of action | User action   | System responses  |
|                                  | 1. System would confirm and notify both actors<br><br>2. System will look at room availability<br><br>3.If rooms are available(see no.4)                | 4. System will upload its details for viewing and booking<br><br>5. Use case exit |
| An alternative course of action  | If rooms available is not updated the system runs back to stage 2 at given time of update   |   |

### 7.3.10. Use Case Update Guest Details on Check out or Fallback

|                                  |  |  |
|----------------------------------|--|--|
| Use Case No.                     | Uc – 10  |  |
| Use Case Name                    | Update Guest details based on checkout or fallback   |  |
| Priority                         | High   |  |
| Actor                            | Hotel administration, System   |  |
| Description                      | This use case allows both system and administration to update Guest details after checkout or sudden day cuts  |  |
| Pre-condition                    | Uc – 06, Uc – 07   |  |
| Post condition                   | Guest details operated on.<br>Eg: If days are cut shorter in concerns to lodging, the field number of days should be updated   |  |
| The fundamental course of action | User action  | System responses   |
|                                  | 1. System or hotel administration gets details from checkout or fall outs of days<br><br>2. System is re-directed to update Guest registration<br><br>3. Administration fills or upon checkout system auto deletes records | 3. Guest registration form is displayed<br><br>4. System prompts for new data<br><br>5. Data is validated<br><br>6. Guest updates made<br><br>7. Upon checkout room is made available for booking again<br><br>8. System confirms update success<br><br>9. Use case exit |
| An alternative course of action  | In 3.1. if more changes are to be made system redirects to stage 3   |  |

### 7.3.11. Use Case Generate Daily Income Report

|                                  |   |   |
|----------------------------------|---|---|
| Use Case No.                     | Uc – 11   |   |
| Use Case Name                    | Generates daily income report of each room according to a specified time  |   |
| Priority                         | High  |   |
| Actor                            | System  |   |
| Description                      | This use case allows daily income reports for all rooms categorized under function, single bed room and double bed room, to be generated at a given time  |   |
| Pre-condition                    | Uc – 01, Uc – 06, Uc – 10, Uc - 12  |   |
| Post condition                   | Daily income reports generated successfully   |   |
| The fundamental course of action | User action   | System responses  |
|                                  | <ol style="list-style-type: none"><li>1. System checks all bookings made and calculates cost according to number of days each guest stays</li><li>2. System retrieves records under all three categories of rooms</li></ol> | <ol style="list-style-type: none"><li>3. Daily income generation is auto loaded at a specific time</li><li>4. Daily income for each room type is generated</li><li>5. Income generation successful</li><li>6. Use case exit</li></ol> |
| An alternative course of action  | If bookings have not been made system will revert to stage 3 at the next days' time given   |   |

### 7.3.12. Use Case Cancel Booking

|                                  |  |   |
|----------------------------------|--|---|
| Use Case No.                     | Uc – 12  |   |
| Use Case Name                    | Cancel booking   |   |
| Priority                         | Medium   |   |
| Actor                            | Guest  |   |
| Description                      | This use case allows the Guest to cancel the booking already booked, due to unavoidable circumstances  |   |
| Pre-condition                    | Uc – 06, Uc – 07, Uc – 08  |   |
| Post condition                   | Guest has successfully cancelled the booking   |   |
| The fundamental course of action | User action  | System responses  |
|                                  | 1. User is on view booking screen where the option for cancel booking is given<br><br>2. User clicks cancel booking button<br><br>5. User clicks confirm cancellation button and affirms request | 3. System responds to cancel booking request<br><br>4. System generates message to confirm cancellation<br><br>6. System cancels booking and displays cancellation successful message<br><br>7. Use case exit |
| An alternative course of action  | If in 4.1. user fails to confirm cancellation the system will re-direct to stage 1   |   |

### 7.3.13. Use Case Logout

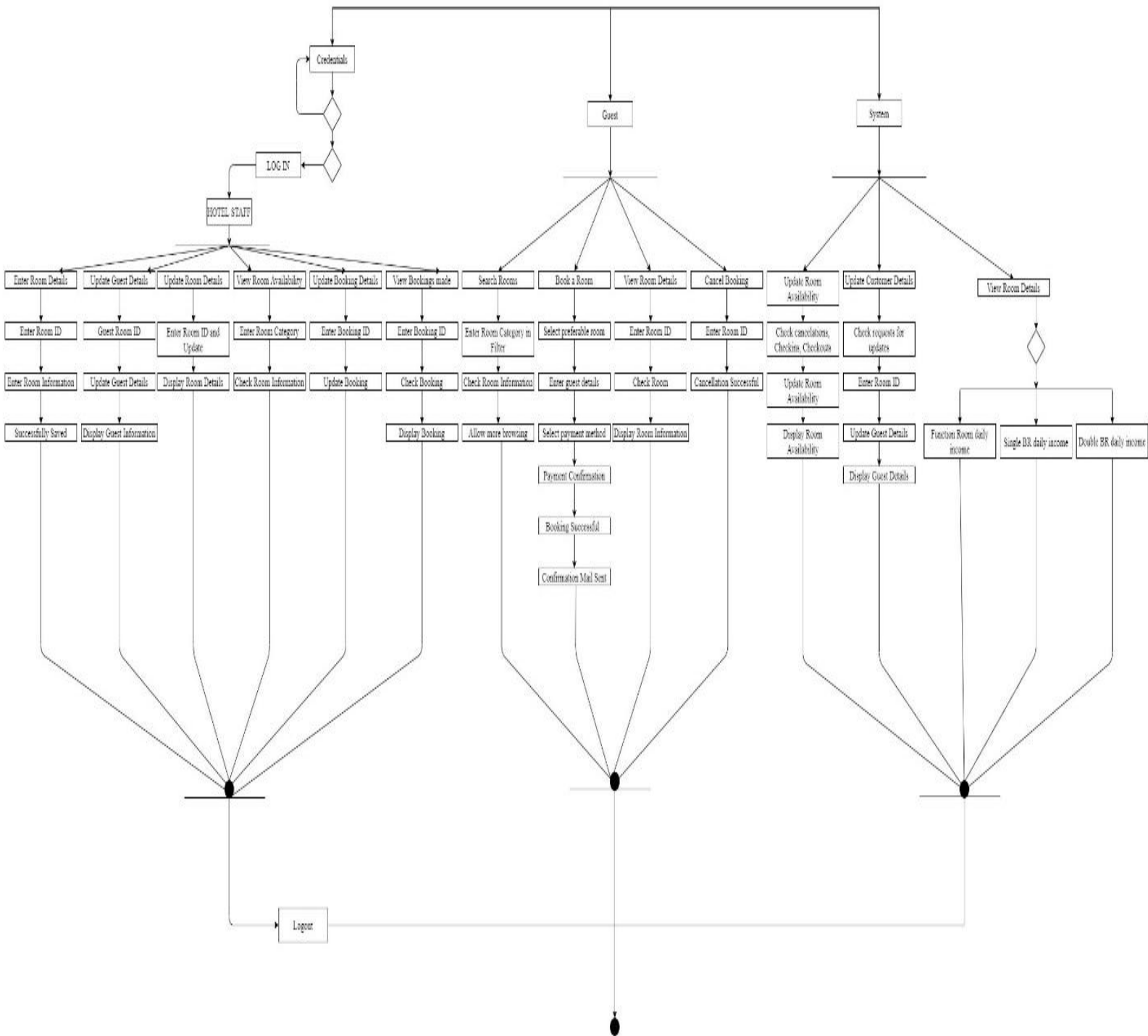
|                                  |  |                                       |
|----------------------------------|--|---------------------------------------|
| Use Case No.                     | Uc – 13  |                                       |
| Use Case Name                    | Logout   |                                       |
| Priority                         | High   |                                       |
| Actor                            | Guest, hotel administration                              |                                       |
| Description                      | This use case allows both actors to logout of the system |                                       |
| Pre-condition                    | Uc – 01  |                                       |
| Post condition                   | Actors have successfully logged out of the system        |                                       |
| The fundamental course of action | User action  | System responses                      |
|                                  | 1. Actors wish to logout                                 | 3. System directs to logout request   |
|                                  | 2. Actors click logout button                            | 4. System logs out                    |
|                                  |  | 5. System displays logged out message |
|                                  |  | 6. Use case exit                      |
| An alternative course of action  | None   |                                       |

## 8. System Analysis and Domain Modeling

### 8.1. Functional View

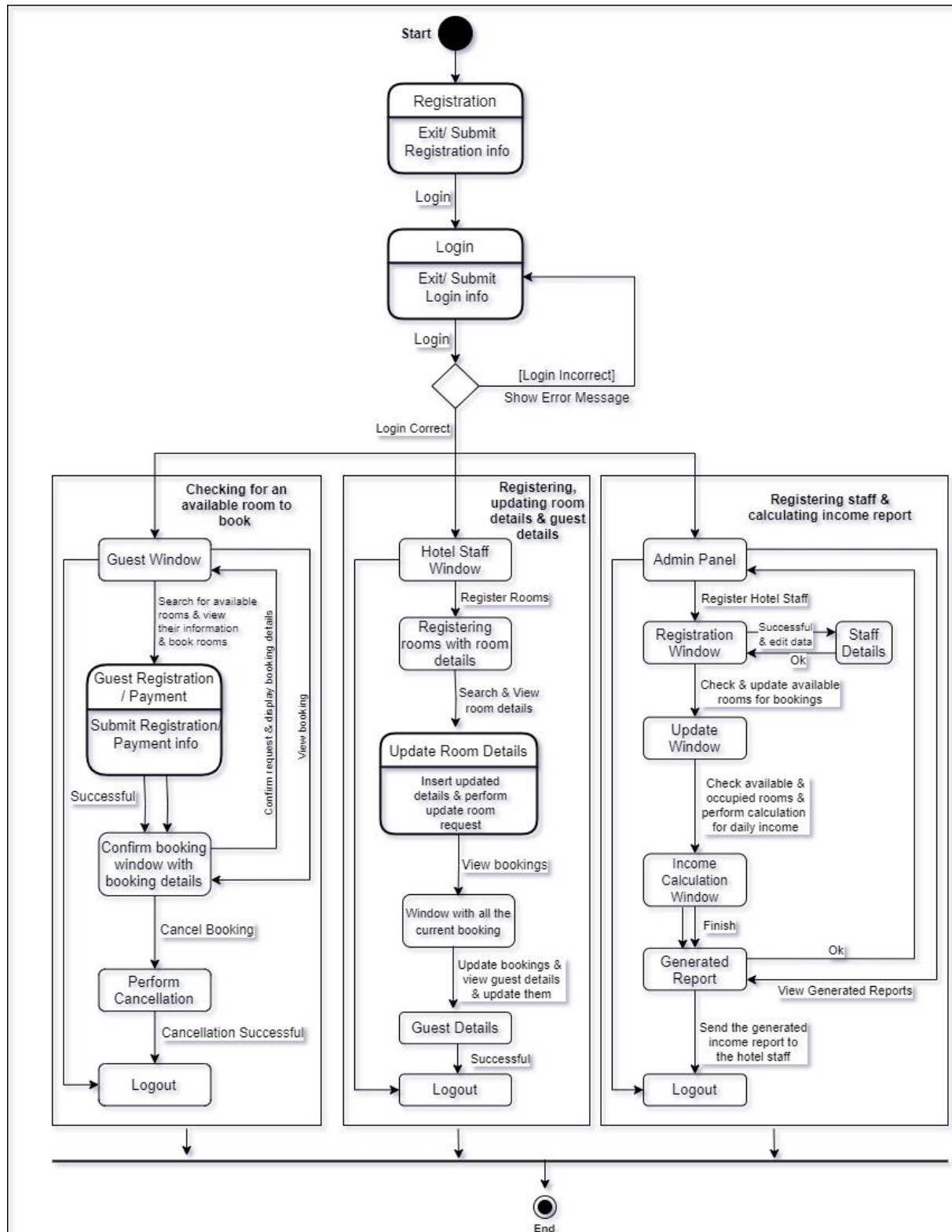
#### 8.1.1. Activity Diagram

Also known as behavioral diagram, the activity diagram in UML is a graphical representation of an executed set of procedures that describes parallel and conditional activities, use cases and functions. Shows the flow of control in the System. (Team, 2020)



### 8.1.2. State machine Diagram

Typically used to describe state dependent behavior of an object, the state machine diagram also known as state chart diagram, is a behavioral diagram in UML that shows transitions between various objects.

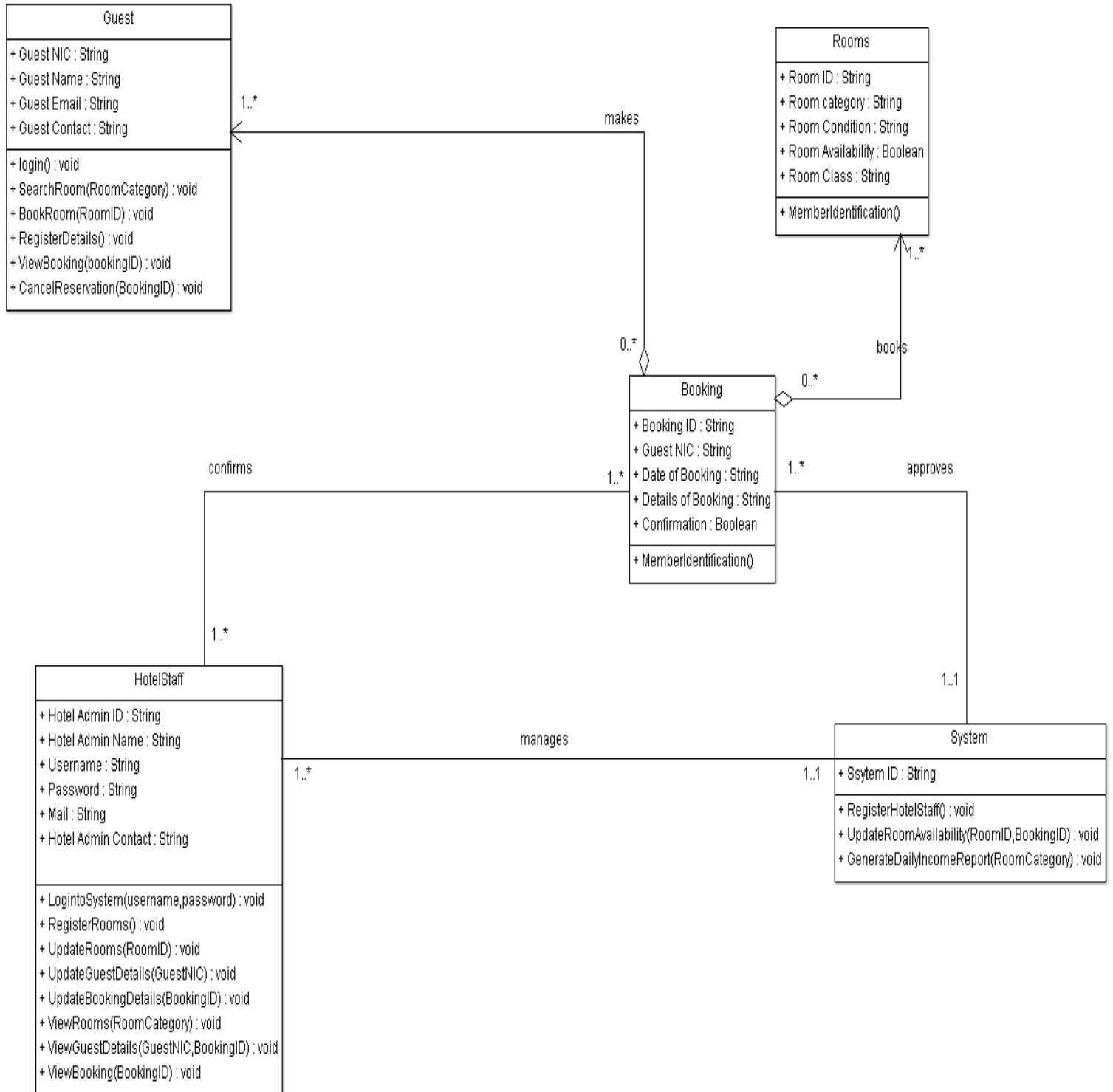




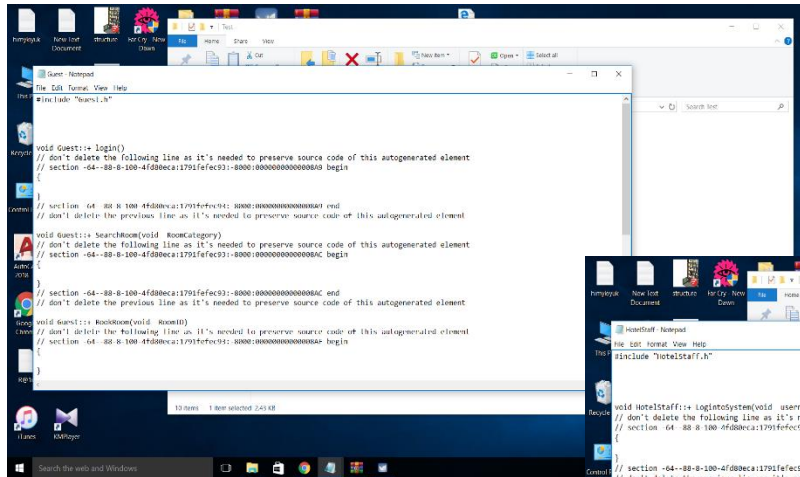
## 8.1. Static View

### 8.1.1. Class Diagram

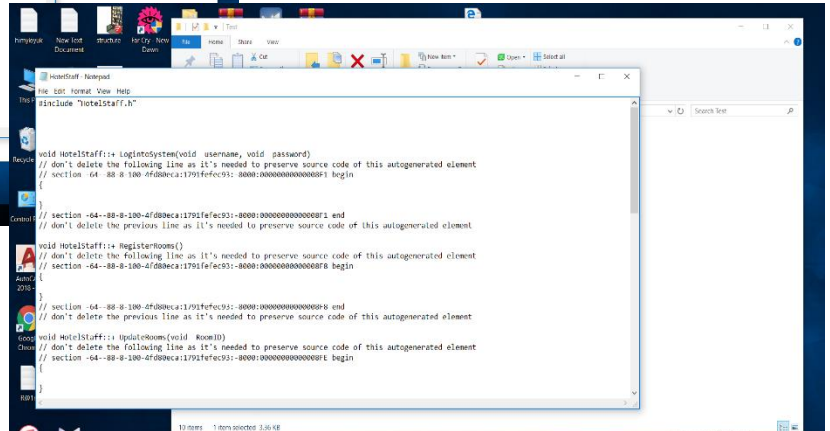
A Class Diagram in UML is a type of Static Structure Diagram describing the structure of the System. In Object Oriented Modelling (OOM), this is known as the leading block.



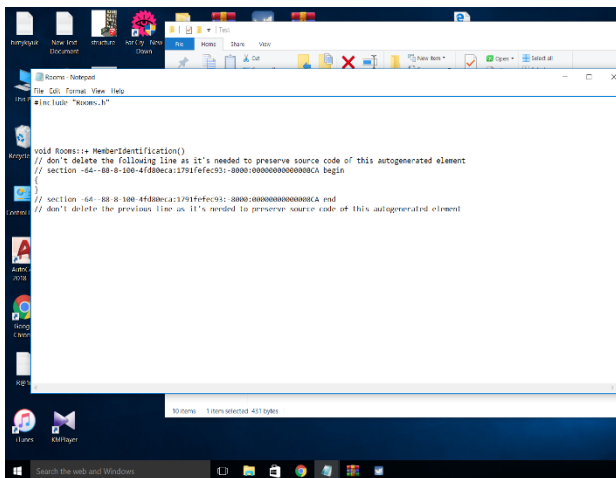
## Code Generation with Argo UML Screenshots as Evidence



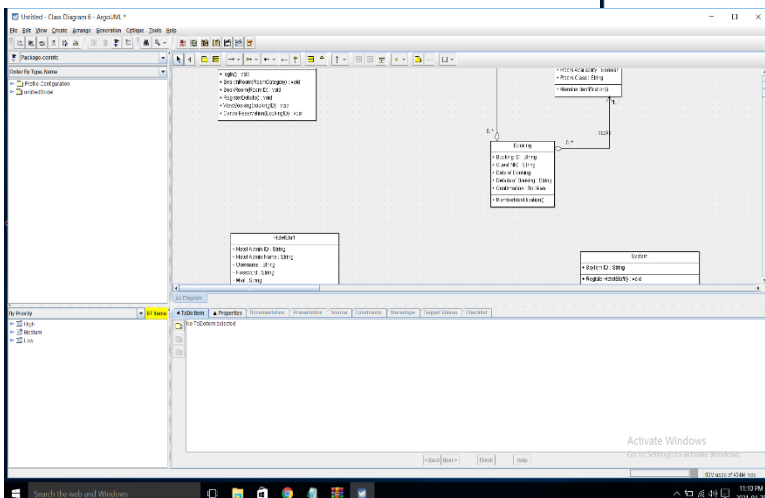
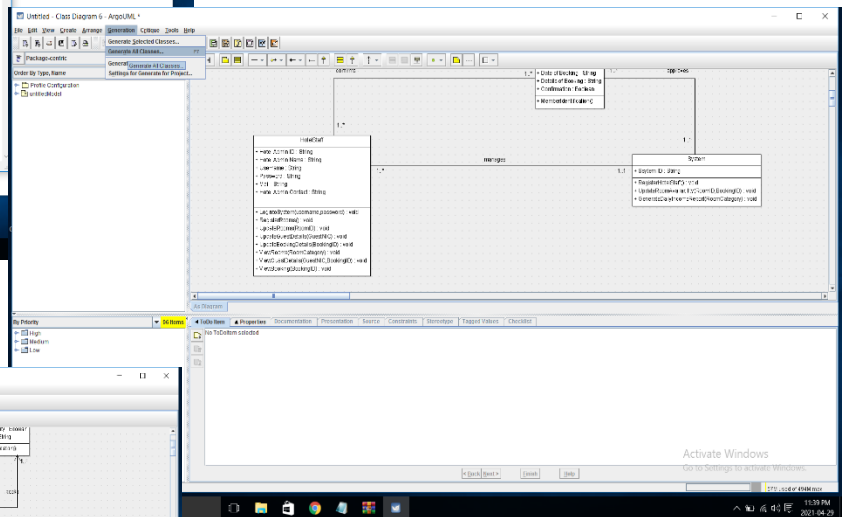
```
void Guest::login()
// don't delete the following line as it's needed to preserve source code of this autogenerated element
// section -64-88-8-100-4f0b6ca1791fec931-8000:0000000000000000 begin
{
// section -64-88-8-100-4f0b6ca1791fec931-8000:0000000000000000 end
// don't delete the previous line as it's needed to preserve source code of this autogenerated element
void Guest::searchRoom(void RoomCategory)
// don't delete the following line as it's needed to preserve source code of this autogenerated element
// section -64-88-8-100-4f0b6ca1791fec931-8000:0000000000000000 begin
{
// section -64-88-8-100-4f0b6ca1791fec931-8000:0000000000000000 end
// don't delete the previous line as it's needed to preserve source code of this autogenerated element
void Guest::bookRoom(void RoomID)
// don't delete the following line as it's needed to preserve source code of this autogenerated element
// section -64-88-8-100-4f0b6ca1791fec931-8000:0000000000000000 begin
{
}
```



```
void HotelStaff::loginSystem(void username, void password)
// don't delete the following line as it's needed to preserve source code of this autogenerated element
// section -64-88-8-100-4f0b6ca1791fec931-8000:0000000000000000 begin
{
// section -64-88-8-100-4f0b6ca1791fec931-8000:0000000000000000 end
// don't delete the previous line as it's needed to preserve source code of this autogenerated element
void HotelStaff::registerRooms()
// don't delete the following line as it's needed to preserve source code of this autogenerated element
// section -64-88-8-100-4f0b6ca1791fec931-8000:0000000000000000 begin
{
// section -64-88-8-100-4f0b6ca1791fec931-8000:0000000000000000 end
// don't delete the previous line as it's needed to preserve source code of this autogenerated element
void HotelStaff::updateRoom(void RoomID)
// don't delete the following line as it's needed to preserve source code of this autogenerated element
// section -64-88-8-100-4f0b6ca1791fec931-8000:0000000000000000 begin
{
}
```



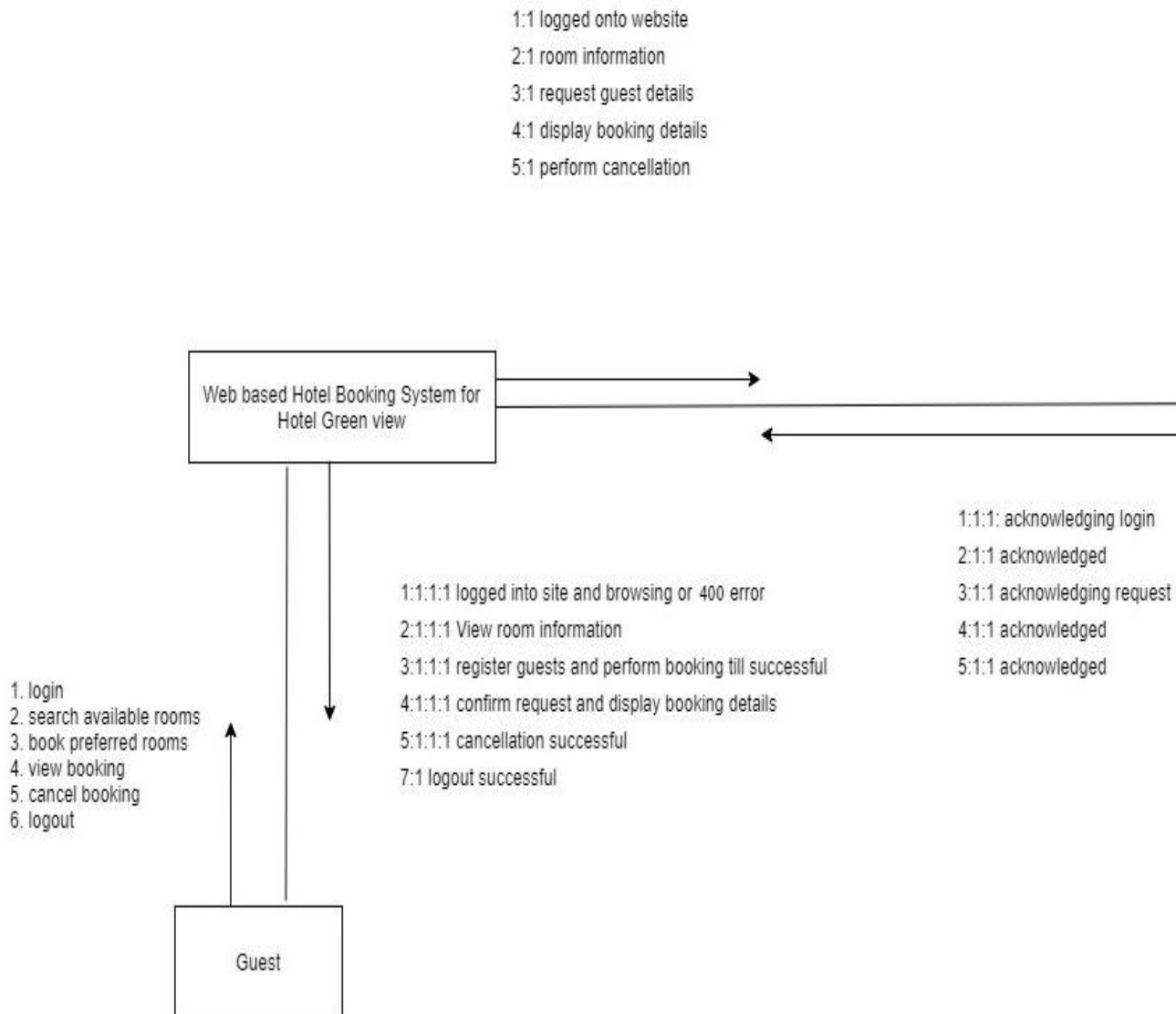
```
void Rooms::MemberIdentification()
// don't delete the following line as it's needed to preserve source code of this autogenerated element
// section -64-88-8-100-4f0b6ca1791fec931-8000:0000000000000000 begin
{
// section -64-88-8-100-4f0b6ca1791fec931-8000:0000000000000000 end
// don't delete the previous line as it's needed to preserve source code of this autogenerated element
}
```



## 8.2. Dynamic View

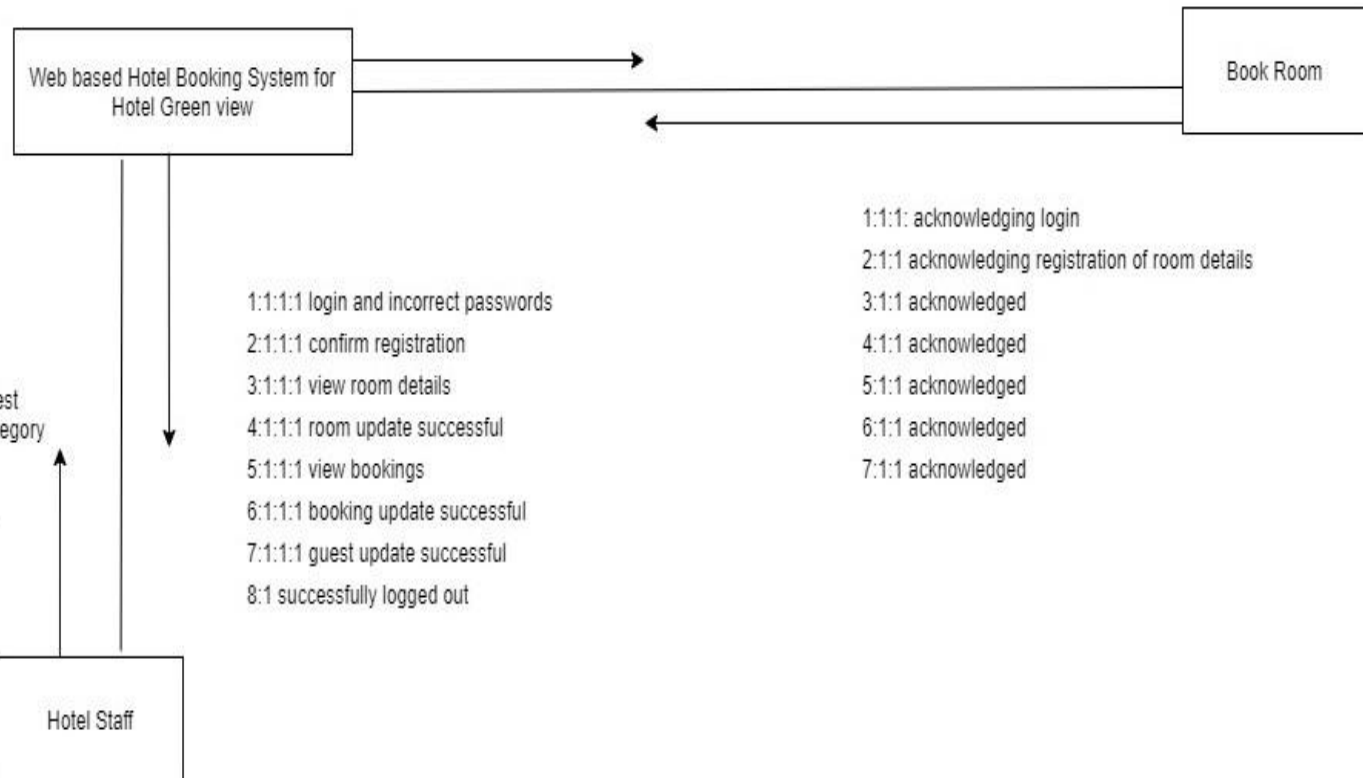
### 8.2.1. Collaboration Diagram

#### Guest

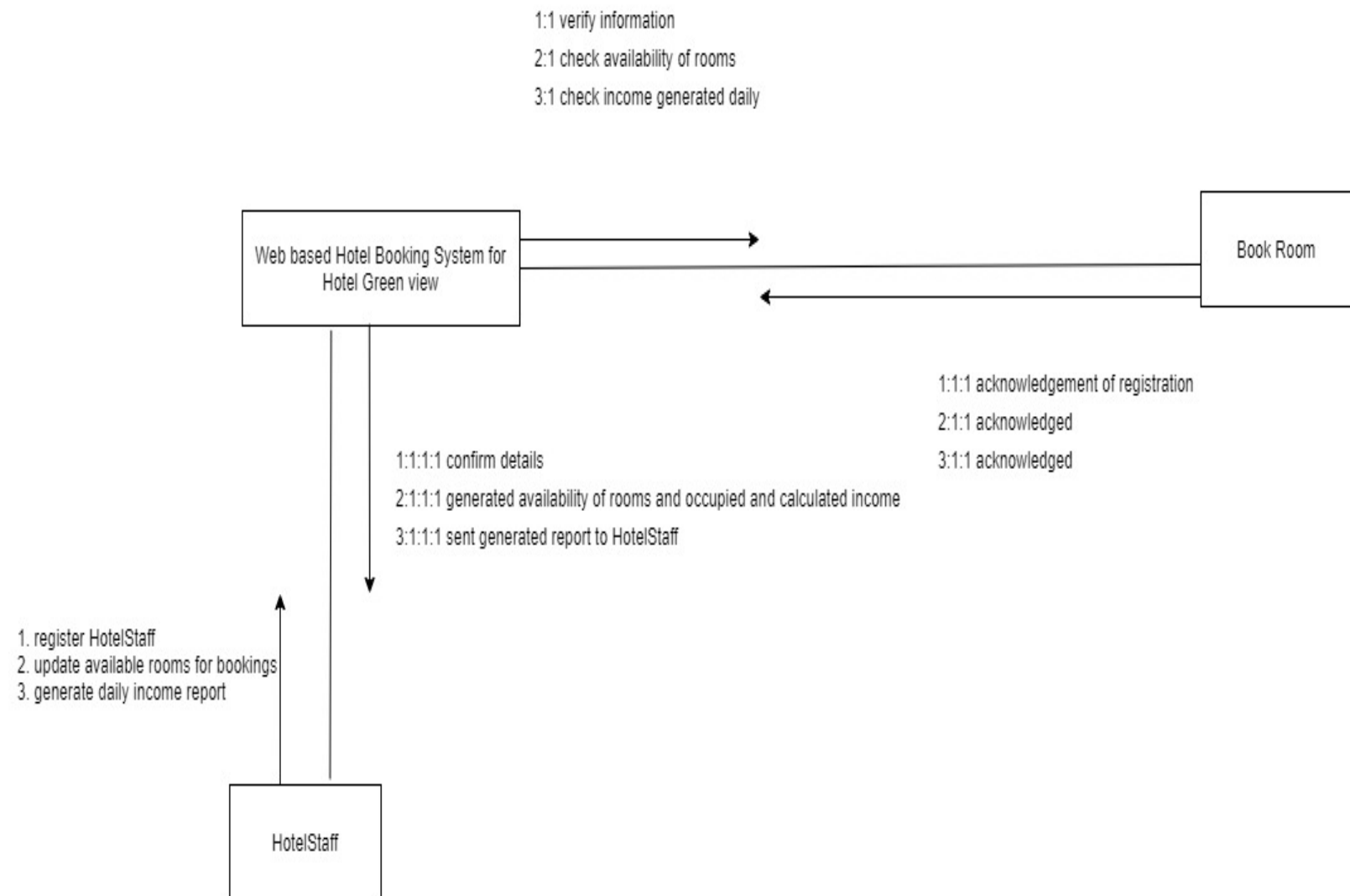


## Hotel Staff

- 1:1 verification of credentials
- 2:1 registering room details
- 3:1 room details
- 4:1 perform update request
- 5:1 perform view bookings for hotel administration
- 6:1 perform update request for bookings
- 7:1 perform update guest list

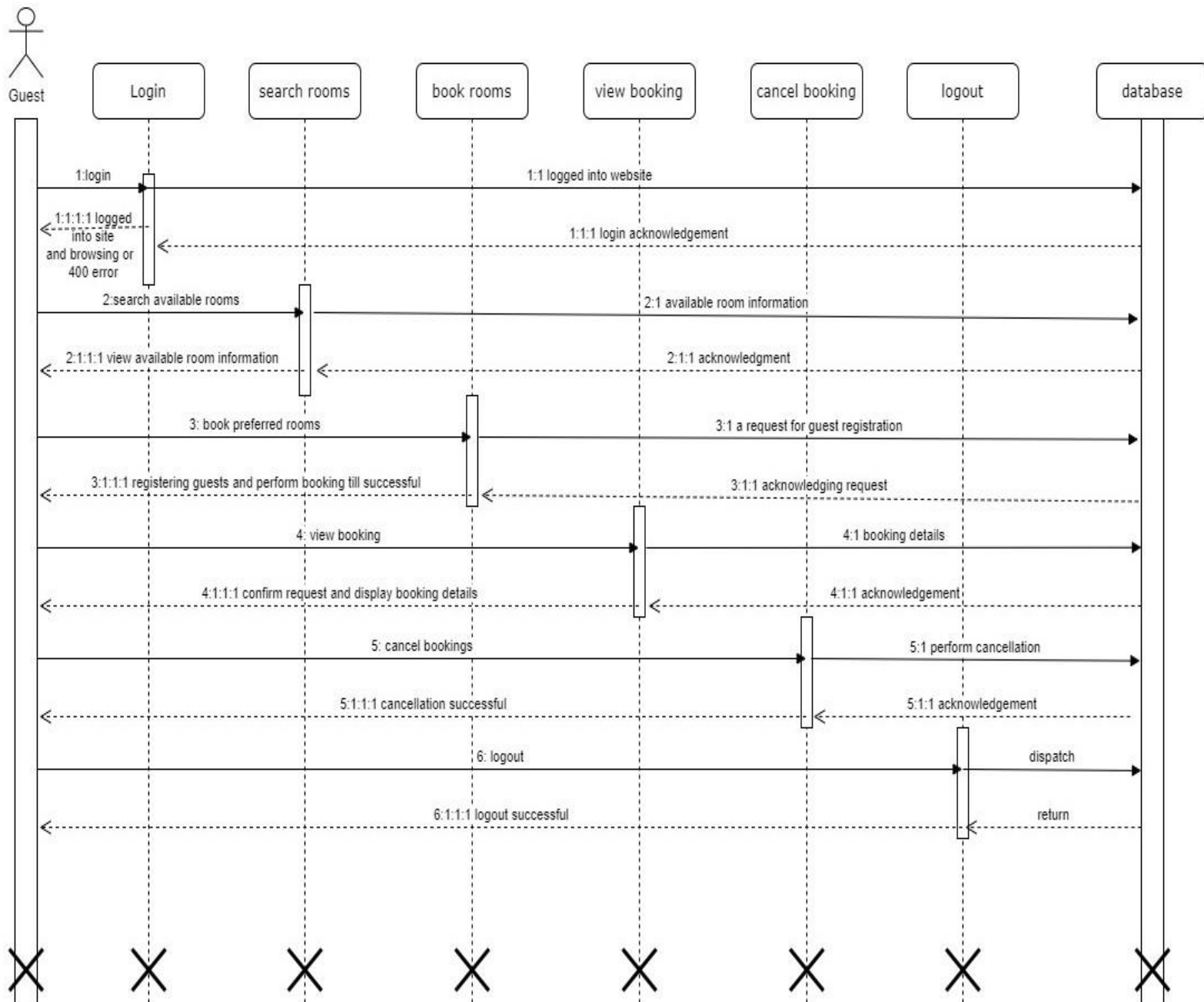


## System

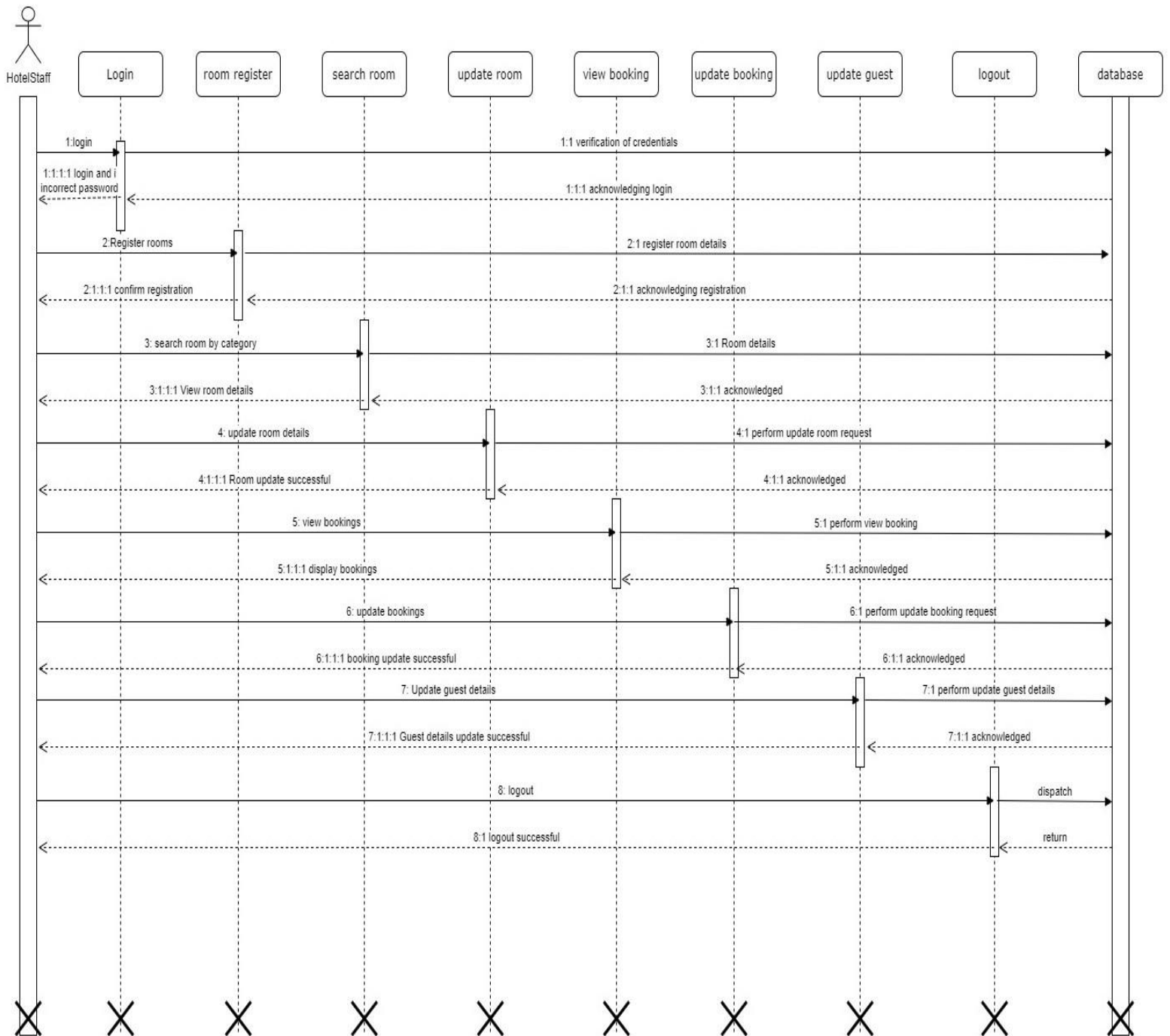


## 8.2.2. Sequence Diagram

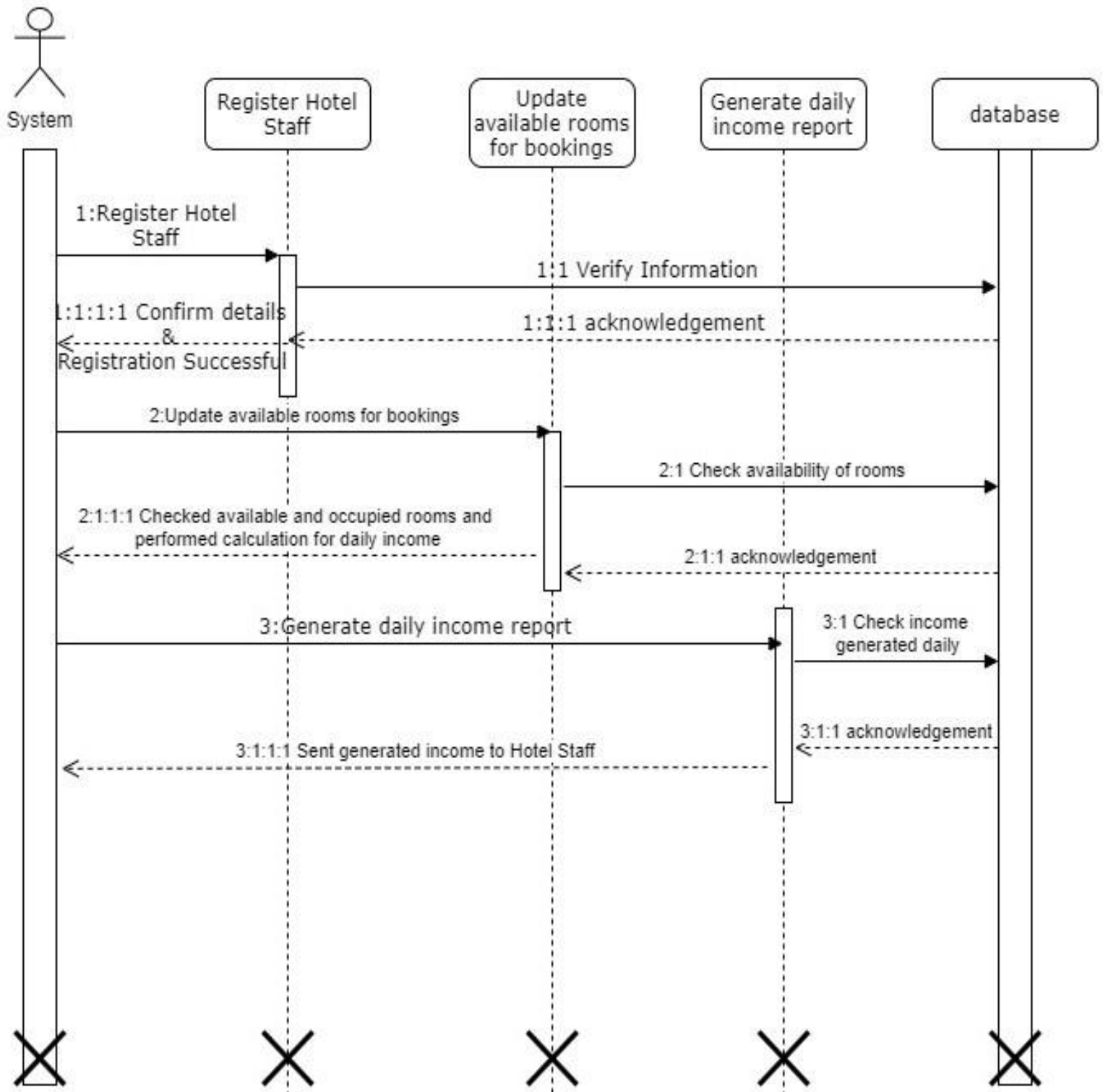
### Guest



## Hotel Staff



## System





## **9. System Design**

### **9.1 Methodology**

When it comes to controlling and managing operations of system that needs to attain a state of achievement within given constraints, a methodology places a key role in system development. The life cycle begins at an analysis, a feasibility study and ends with a solution to counteract the problem. (Vogt, 2019) (Bautista, 2010)

### **9.2 Analysis of a traditional Hotel booking System**

When manually handling operations in relation to booking, a Guest would have to personally voice out an order of reservation either through a phone call or a visit. There would be no proper information on the hotel and its facilities either which could easily lead to misunderstanding. The staff would keep information in either ledgers or spread sheets. Chances of making mistakes, misplacing or losing data is high. A Guest can be under charged or over taxed according to miscalculations made in report generation.

### **9.3 Design for the proposed system**

The booking system to be developed is configured to carry out the requirements as listed below.

1. Making online bookings – This system will increase efficiency of the process of receiving rooms online fulfilling Guest satisfaction by presenting a catalog of room available under the categories function halls, single BR's, double BR's.
2. Managing Hotel Staff and Guest Registrations - The software solution would allow the actors to register personal details and create accounts with varied access authentication.
3. Cancellation of bookings – The system would allow guests to allow cancel bookings before checking in, updating chosen booked available for booking again.
4. Updating room detail properties – During holiday spikes or seasonal offers if the hotel wishes to add discounts to payment of rooms, operation functions for said purpose is made available. (Gupta, 2020)
5. Generating daily income reports – At the end of the day at a given time already saved, a report on income made, under all three categories of booking would be generated.
6. Once booked, confirmation email sent – Once a guest has booked a preferred room, and it is registered in the database, the system would automatically send a confirmation mail with related details needed when checking in. (foreUP, 2018) (Team T. E., 2018)

## **9.4 Users and their Characteristics**

### **9.4.1 System / Super Admin**

- Update available free rooms
- Generate booking confirmed email
- Cancelling booking upon check out
- Generating daily income report

### **9.4.2 Hotel Administration (Hotel staff)**

- Login into system
- Updating information of rooms to the database
- Cancelling bookings manually due to unseen circumstances
- Maintain personal information of guests
- Confirms booking
- Giving information of facilities available
- Checking conditions of Rooms

### **9.4.3 Guests (Guest)**

- Login into system
- Visiting Hotel Website
- Referring rooms and facilities available
- Booking rooms
- Making payment
- Cancel booking

## **9.5 Workflow of the proposed system**

### **9.5.1 Log in**

1. The system should enable the actors (guests / hotel staff) to login relevantly.
  - Hotel staff – log in with their relevant valid credentials.
  - Guests – log in online at free will
2. The system allows the Super Admin to create new accounts for registration of newly hired staff in charge of ushering guests and management.
3. The system would allow logged in staff to change password at will.
4. The system would allow the users to log out at will.

### **9.5.2 Bookings**

5. Guests can view rooms they want to book according to category.
6. Guest can view selected room description.
7. The system would allow Guests to get registered at the time of booking.
8. System will update availability of rooms based on check – ins and check – outs.
9. System allows guests to view the bookings they have made.
10. Guests can cancel bookings at any time using a confirmation code received at the time booking.
11. Hotel Staff would be able to see bookings made as of new.
12. System provides unique numbers to act as primary keys for each reservation made.
13. Guests can update their registration if needed.
14. System will provide a payment option (offline/online).
15. System sends confirmation email at time of booking.

### **9.5.3 Rooms (function, single BR, and double BR)**

16. System allows staff administration to update or register details of new rooms.
17. System allows searching of rooms according to specifications.
18. System displays all available rooms for booking.
19. System displays booked rooms and details of client.

### **9.5.4 Reports generated**

20. System Checks each room booked under all three categories and generates reports.
21. System checks available rooms for booking and generates reports.
22. System generates a daily income report at specified time according to the number of rooms looked.
23. System generates printable statistic report of profit and expenditure.

## 9.6 Implementation Requirement

This phase focuses on the development of the proposed system and how it is to be implemented.

1. We need to focus on coding and choosing a preferable get advantages programming language.
2. We need to consider Verification and validation.
3. We need to consider User Interaction. (Meng, 2017)

When considering point 1, we need to choose a language carefully. It is the language that will define the constraints and complexity of the system. PHP, MySQL, HTML, CSS are some of the best options you could take. (Mareike, 2017)

Advantages that follow when using PHP

- PHP is a language that covers almost every requirement in the system.
- It is a scripting language that will run on any device regardless of OS (operating system and is usually used to developed dynamic websites).
- The same follows with MySQL databases.
- Learning opportunity.

## 9.7 System requirements

The following focuses on Software and Hardware Requirements for the Software solution.

### 9.7.1 Hardware specification

This system deals with real time updates, intensive calculations, and massive volumes of data transactions. For this a good internet connection is needed. Following are,

- LAN / WAN devices.
- Printers, Digital Cameras.
- Internet service Providers.
- Computer Workstations.
- RAID products as such

### 9.7.2 Software Specifications

1. Operating System (windows/LINUX/MAC OS).
2. XAMPP / WAMPP Server.
3. MySQL Server

## 10. Conclusion

### 10.1. Individual Contribution

| Name and Index Number         | Contribution  |
|-------------------------------|---|
| K.C. Berenger - 10673083      | I drew the Use Case Diagram and Use case Descriptions. I also helped in getting the documentation done. I drew the Class diagram and helped with the Sequence diagrams. |
| I.A. Dissanayake - 10673116   | I drew the Sequence Diagrams and noted down their descriptions. I also helped in analyzing the manual systems of the proposed system in comparison to it.               |
| M.D. Dissanayake - 10673118   | I drew collaboration diagram and helped in writing the introduction to the documentation.   |
| S.S.D.H. Fernando - 10673123  | I helped with drawing the collaboration diagram and gave ideas for system design.   |
| P.G.G.M.B. Bandara - 10673076 | I designed and drew the activity diagram and noted down the scope and objectives.   |
| M.A.O.G. Kithmina - 10673260  | I helped with the documentation by searching and writing system requirements, implementation requirements, methodology and workflow for the proposed system.            |

## 10.2. References

- Bautista, N. (2010). *A Beginner's Guide to Design Patterns*.
- Edwards, A. (2020). *How to Book a Hotel Room*.
- foreUP, T. (2018). *Advantages and Disadvantages of Using an Online Booking System*.
- Gupta, V. (2020). *A Beginner's Guide to Design Patterns*.
- Gupta, V. (2020). *How to Create An Online Booking System In 30 Minutes*.
- Kupe Kupersmith, P. M. (n.d.). *How to Create Use Case Description for Your Business Analysis Report*.
- Larson, E. |. (2004). *Use cases: what every project manager should know*.
- Mareike. (2017). *12 most important features your online booking system should have*.
- Meier, H. (2017). *First time using draw.io: It's easy to use!*
- Meng, C. L. (2017). *Design and Implementation of Online Booking System of University Sports Venues*.
- Team, S. (2020). *Everything you need to know to make a great UML diagram*.
- Team, T. E. (2018). *The 7 Most Important Software Design Patterns*.
- Vogt, P. (2019). *Patterns in software design*.
- Walker, H. M. (2005). *An Introduction to ArgoUML*.

\*\*\*