Requirements Engineering

HotelReservation\_SYS

Submitted By: Mark O Donoghue (t00196848)

Computing with Software Development

Date Submitted: 03/05/2018

**Table of Contents**

[1. Introduction/overview 4](#_Toc500864014)

[2. Functional Components 5](#_Toc500864015)

[3. User Requirements 6](#_Toc500864016)

[4. System Requirements 7](#_Toc500864017)

[4.1. System Level Use Case Diagram 7](#_Toc500864018)

[4.2. Manage Rooms 8](#_Toc500864019)

[4.2.1. Add Room Rate 8](#_Toc500864020)

[4.2.2. Update Room Rate 10](#_Toc500864021)

[4.2.3. Add Room 13](#_Toc500864022)

[4.2.4. Update Room 15](#_Toc500864023)

[4.2.5. Close/Open Room 17](#_Toc500864024)

[4.3. Manage Reservations 19](#_Toc500864025)

[4.3.1. Make Reservation 19](#_Toc500864026)

[4.3.2. Cancel Reservation 24](#_Toc500864027)

[4.4.3. Check In 26](#_Toc500864028)

[4.4.4 Check Out 29](#_Toc500864029)

[4.4. Perform Admin 31](#_Toc500864030)

[4.4.1. Generate Sales Analysis 31](#_Toc500864031)

[4.4.2. Generate Rooms Analysis 33](#_Toc500864032)

[5. System Model 36](#_Toc500864033)

[5.1. Level-0 DFD 36](#_Toc500864034)

[5.2. Level-1 DFD 37](#_Toc500864035)

[5.3. Level-2 DFD (Manage Rooms) 38](#_Toc500864036)

[5.4. Level-2 DFD (Manage Reservations) 39](#_Toc500864037)

[5.5. Level-2 DFD (Perform Admin) 40](#_Toc500864038)

[6. Data Model (Class Diagram) 41](#_Toc500864039)

[6.1. Class Diagram 41](#_Toc500864040)

[6.2. Relational Schema 42](#_Toc500864041)

[6.3. Database Schema 42](#_Toc500864042)

[7. Conclusion 44](#_Toc500864043)

[8. Appendices 45](#_Toc500864044)

[8.1. Appendix A – Title 45](#_Toc500864045)

[8.2. Appendix B – Title 45](#_Toc500864046)

# Introduction/overview

HotelReservation\_SYS was designed with the intention of being a viable software option for any hotel in the modern climate. I designed HotelReservation\_SYS to perform functions such as room administration, reservation administration, rate administration as well as generating detailed sales/rooms reports which I felt a vital to any established hotel.

Primary functions of this system were adding and updating rates, adding updating and closing rooms dealing with all aspects of reservation and checkIn and generating detailed sales reports as well as detailed reports on how rooms were performing.

# Functional Components

# User Requirements

1. HotelReservation\_SYS will perform room administration
   1. HotelReservation\_Sys will register a room rate
   2. HotelReservation\_SYS will register a new room.
   3. HotelReservation\_Sys will update an existing room.
   4. HotelReservation\_Sys will close an existing room.
2. HotelReservation\_SYS will perform reservation administration
   1. HotelReservation\_SYS will make a reservation.
   2. HotelReservation\_SYS will cancel an existing reservation.
   3. HotelReservation\_SYS will check in a customer.
   4. HotelReservation will check out a customer.
3. HotelResrvation\_SYS will perform general administration
   1. HotelResrvation\_SYS will generate a sales analysis report.
   2. HotelReservation\_SYS will generate a rooms analysis report.

# System Requirements

|  |  |  |
| --- | --- | --- |
| **Functional Requirements** | **Non-Functional Requirements** | **Domain Requirements** |
| 1. To log a room rate to be added to allow for pricing on rooms offered 2. To allow rooms to be created updated or closed depending on the needs of the managers 3. To allow the user to make future reservations update and cancel them. 4. To store all inputs and statistics on a database 5. The system will allow the users to generate detailed reports in the area of sales and popularity of product | 1. Speed is key. There will be minimal windows. Each entry will happen on one form. 2. Security is of the utmost importance. Each staff member who has been trained will have their own unique password that will be randomly generated and can be re-assigned if they fear the password has been compromised. 3. Design will be minimalistic. HotelReservation\_SYS will be as visually pleasing as it is functional. | 1. Due to security issues the software will not have online capability however the Manager will have access at home. The system will only be available in the office network. |

## System Level Use Case Diagram

HotelReservatiom\_SYS will require a room rate from the Manager. This will allow a type to be assigned to any room added by the manager. The manager will then have the ability to update or close/open a room as he sees fit.

HotelReservation\_SYS will allow a receptionist to make a reservation. This function will also include an option for updating and cancelling said reservations. There will also be a check In and check Out function. The check out feature will allow for payment of services.

HotelReservation\_SYS will allow a manager to generate a sales report. The system will also allow for a manager to get a detailed room analyse report.

HotelReservation\_SYS

Owner/ Manager

Receptionist

## Manage Rooms

This component will register a room rate and add new rooms. The component will also allow the user to update existing rooms and close them.

### Add Room Rate

HotelReservation\_SYS will register a room rate.

Manager

«includes»

<<extends>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Add Room Rate** | |
| **Use Case Id** | 4.2.1 | |
| **Priority** | High | |
| **Source** | Manager | |
| **Primary Business Actor** | Manager | |
| **Other Participating Actors** | N/A | |
| **Description** | The purpose of this function is to allow the user to register a Room Rate into the HotelReservation\_SYS System | |
| **Preconditions** | The room rate must consist of 3 letters and be unique | |
| **Trigger** | N/A | |
| **Expected Scenario** | **Receptionist** | **System Response** |
|  | **Step 1: The Manager invokes the Add Room Rate Function.**  **Step 3: The Manager enters the room rate details:**   * **Room\_Type** * **Description** * **Rate** | **Step 2: The system displays the UI.**  **Step 4: The system validates the Room Rate details:**   * **All fields must be entered** * **Type cannot exceed 3 characters** * **Rate must be only decimal value** * **Room\_Type must be unique** * **Field only accepts letters**   **Step 5: Save Room Rate details in the Room\_Rate file:**   * **Room\_Type** * **Description** * **Rate**   **Step 6: Display Confirmation Message**  **Step 7: Clear The UI** |
| **Alternate Scenarios** | **Manager** | **System Response** |
| **Field not entered** |  | **Step 4: A blank field entered**  **Step 5: Display message “This Field must be entered”**  **Step 6: Position cursor in offending field and return to step 3** |
| **Type Data exceed 3 chars** |  | **Step 4: Type field exceeds 3 characters.**  **Step 5: Display Message “This Field cannot exceed 3 characters”**  **Step 6: Position cursor in Room type field and return to step 3** |
| **No Numeric Data detected** |  | **Step 4: A non-decimal value is entered**  **Step5: Display message” Rate must be a decimal value”**  **Step6: Position cursor in the offending field and return to step 3** |
| **Room\_Type must be unique** |  | **Step 4: A previously used Room\_Type has been entered**  **Step 5: Display message “This Room Type is already in existence”**  **Step 6: Position cursor in the offending field and return to step 3** |
| **Field only accepts letters** |  | **Step 4: A Numeric Value is entered into room-type or description**  **Step 5: Display Message “This textbox only accepts letters”**  **Step 6: Position cursor in the offending field and return to step 3** |
| **Conclusions** | Room Rate is registered | |
| **Post conditions** | Add a room and update a room can now be assigned a room rate | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### 4.2.2. Update Room Rate

Hotel Reservation\_SYS will update a pre-existing room rate

Manager

«includes»

<<extends>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Update Room Rate** | |
| **Use Case Id** | 4.2.1 | |
| **Priority** | High | |
| **Source** | Manager | |
| **Primary Business Actor** | Manager | |
| **Other Participating Actors** | N/A | |
| **Description** | The purpose of this function is to allow the user to update a Room Rate into the HotelReservation\_SYS System | |
| **Preconditions** | The room rate must already have been made in order for the manager to update it | |
| **Trigger** | N/A | |
| **Expected Scenario** | **Manager** | **System Response** |
|  | **Step 1: The Manager invokes the Update Room Rate Function.**  **Step 5: The Manager selects the Room\_Type he wishes to alter from the list of previously added Room\_Types.**   * **Room\_Type**   **Step 7: The Manager can then update the following details:**   * **Description** * **Rate** | **Step2: System validates that there is already a room rate in existence to be updated**   * **RoomRate not in existence**   **Step 4: The system retrieves the pre-existing Room\_Type and description from the Room\_Rate file and displays the UI**  **Step 6: The system then retrieves the rate from the Room\_Rate file and displays the UI. Filling in the description and rate textboxes with the description and rate that is associated with the chosen Room Type**  **Step 8: The system validates the Room Rate details:**   * **All fields must be entered** * **Rate must be only decimal values**   **Step 9:The System updates details in the Room\_Rate file:**   * **Description** * **Rate**   **Step 10: Display Confirmation Message**  **Step 11: Clear The UI** |
| **Alternate Scenarios** | **Manager** | **System Response** |
| **RoomRate not in existence** | **Step 1: Manger invokes the update Room Rate function** | **Step 2: System finds that no Room Type has been entered into the system yet and displays the error message “There are no room rates available for Update”**  **Step 3: System clears the UI and return to Step 1** |
| **Field not entered** |  | **Step 8: A blank field entered**  **Step 9: Display message “This Field must be entered”**  **Step 10: Position cursor in offending field and return to step 7** |
| **No Numeric Data detected** |  | **Step 8: A non numeric value is entered**  **Step 9: Display message ”This field must be numeric”**  **Step 10: Position cursor in offending field and return to step 7** |
| **Conclusions** | A Room Rate is now updated | |
| **Post conditions** |  | |
| **Business Rules** | A Room\_Type must already exist in order to be updated. A new Room\_Type cannot be made in the update Room Rate function | |
| **Implementation Constraints** |  | |

### 4.2.3. Add Room

HotelReservation\_SYS will register a room.

Manager

«includes»

<<extends>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Add Room** | |
| **Use Case Id** | 4.2.3 | |
| **Priority** | High | |
| **Source** | Manager | |
| **Primary Business Actor** | Manager | |
| **Other Participating Actors** | N/A | |
| **Description** | The purpose of this function is to allow the user to register a room into the HotelReservation\_SYS System | |
| **Preconditions** | There must be a room available to facilitate the room being added.  A Room rate must be added before you can add a room | |
| **Trigger** | N/A | |
| **Expected Scenario** | **Manager** | **System Response** |
|  | **Step 1: The Manager invokes the Add Room Function.**  **Step 4: The Manager enters the add room details:**   * **Room\_No**   **And then selects the rate he wishes to apply to the room from a list:**   * **Room\_Type** | **Step2: System validates that there is already a room rate in existence to be updated**   * **RoomRate not in existence**   **Step 3: The system retrieves the available Room\_Type from the room rates file and displays the UI.**  **Step 5: The system validates the Room details:**   * **All fields must be entered** * **Room\_No must consist of numeric values** * **Room\_No must be unique cannot use a pre - existing Room No**   **Step 6: System automatically assigns Status a value of “A”(available).**  **Step 7: Save Add Room details in the Rooms file:**   * **Room\_No** * **Room\_Type** * **Status**   **Step 8: Display Confirmation Message**  **Step 9: Clear The UI** |
| **Alternate Scenarios** | **Manager** | **System Response** |
| **RoomRate not in existence** | **Step 1: Manger invokes the add Room function** | **Step 2: System finds that no RoomRate has been entered into the system yet and displays the error message “A rate has not been entered please add a roomrate before adding a”**  **Step 3: System clears the UI and return to Step 1** |
| **Field not entered** |  | **Step 5: A blank field entered**  **Step 6: Display message “This Field must be entered”**  **Step 7: Position cursor in offending field and return to step 4** |
| **Room\_No Data must be numeric** |  | **Step 5: The Room\_No field must be numeric values.**  **Step 6: Display Message “This Field must consist of numeric values”**  **Step 7: Position cursor in Room\_No field and return to step 4** |
| **Cannot use an existing Room No** |  | **Step 5: Room\_No already in use**  **Step 6: Display Message “This room already exists please try to re-enter a valid Room\_No”**  **Step 7: Position Cursor in Room\_No field and return to step 4** |
| **Conclusions** | Add Room is registered | |
| **Post conditions** | A Room is now registered in the HotelReservation\_SYS System and is available to be booked | |
| **Business Rules** | A Room is not allowed to be registered unless one or more room rates have been registered. | |
| **Implementation Constraints** |  | |

### 4.2.4. Update Room

HotelReservation\_SYS will update an existing room.

Manager

«includes»

<<extends>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Update Room** | |
| **Use Case Id** | 4.2.4 | |
| **Priority** | High | |
| **Source** | Manager | |
| **Primary Business Actor** | Manager | |
| **Other Participating Actors** | N/A | |
| **Description** | The purpose of this feature is to allow the user to update a pre-existing room. | |
| **Preconditions** | The room must have already been created in the add room function before you can update it. | |
| **Trigger** | N/A | |
| **Expected Scenario** | **Manager** | **System Response** |
|  | **Step 1: The Manager invokes the Update Room Function.**  **Step 4: The Manager then selects the specific field from a list:**   * **Room\_No**   **Step 6: The Manager then alters the Room\_Type from a list:**   * **Room\_Type**   **And presses the update button** | **Step 2: System validates that there is already a room created before allowing you to update**   * **Room not in existence**   **Step 3: The system retrieves the all available Room\_No from the rooms file and displays the UI.**  **Step 5: The system then retrieves The Room\_Type associated with that Room\_No from the rooms file and all available Room\_Types from the Room Rate file and displays it in the UI**  **Step 7: The system validates the Room details**   * **Room has a reservation**   **Step 8: Update Room details in the Rooms file:**   * **Room\_No** * **Room\_Type**   **Step 9: Display Confirmation Message**  **Step 10: Clear The UI** |
| **Alternate Scenarios** | **Manager** | **System Response** |
| **Room not in existence** | **Step 1: Manger invokes the update Room function** | **Step 2: System finds that no Room has been entered into the system yet and displays the error message “There are no rooms available for update”**  **Step 3: System clears the UI and return to Step 1** |
| **Room has a reservation** |  | **Step 7: Room has a reservation made on it**  **Step 8: Display the message “This room has a reservation pending cannot update”**  **Step 9: Position Cursor in Room\_No field and return to step 6** |
| **Conclusions** | Room is successfully updated | |
| **Post conditions** |  | |
| **Business Rules** | Room cannot be updated if it is has a future reservation associated with it | |
| **Implementation Constraints** | Rooms with future reservations are not eligible for being updated | |

### 4.2.5. Close/Open Room

HotelReservation\_SYS will allow a user to close and reopen existing rooms

Manager

«includes»

<<extends>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Close/Open Room** | |
| **Use Case Id** | 4.2.5 | |
| **Priority** | High | |
| **Source** | Manager | |
| **Primary Business Actor** | Manager | |
| **Other Participating Actors** | N/A | |
| **Description** | The purpose of this feature is to allow the user to close a pre-existing room and to allow a closed room to return to its status as available. | |
| **Preconditions** | The room must have already been created in the add room function before you can close it. | |
| **Trigger** | N/A | |
| **Expected Scenario** | **Manager** | **System Response** |
|  | **Step 1: The Manager invokes the Close/Open Room Function.**  **Step 4: The Manager then selects the specific Room from a list:**   * **Room\_No**   **Step 6: The Manager then alters the Status from a list he can change this from “A” (available) to “C” (closed) or “C” to “A”:**   * **Status**   **Manager then selects close/open button** | **Step 2: System validates that there is already a room created before allowing you to update**   * **Room not in existence**   **Step 3: The system retrieves all available Room\_No from the rooms file and displays the UI**  **Step 5: The system then retrieves The Status associated with that Room\_No from the rooms file and displays it in the UI**  **Step 7: The system validates the Room details**   * **Room has a reservation**   **Step 8: Update Room details in the Rooms file:**   * **Room\_No** * **Status**   **Step 9: Display Confirmation Message**  **Step 10: Clear The UI** |
| **Alternate Scenarios** | **Manager** | **System Response** |
| **Room not in existence** | **Step 1: Manager invokes the Close/Open function** | **Step 2: System finds that no Room has been entered into the system yet and displays the error message “There are currently no rooms available for their status to be changed”**  **Step 3: System clears the UI and return to Step 1** |
| **Room has a reservation** |  | **Step 7: Room has a reservation made on it**  **Step 8: Display the message “This room has a reservation pending cannot open/close”**  **Step 9: Position Cursor in Room\_No field and return to step 6** |
| **Conclusions** | Room is successfully closed | |
| **Post conditions** | N/A | |
| **Business Rules** | A room must already have been added in order to be closed.  Room cannot be updated if it has a future reservation associated with it | |
| **Implementation Constraints** | N/A | |

## Manage Reservations

This component will allow the user to make a reservation/ cancel a reservation. This component will also allow for the check-in/checkout of a customer.

### Make Reservation

HotelReservation\_SYS will allow a user to make a reservation.

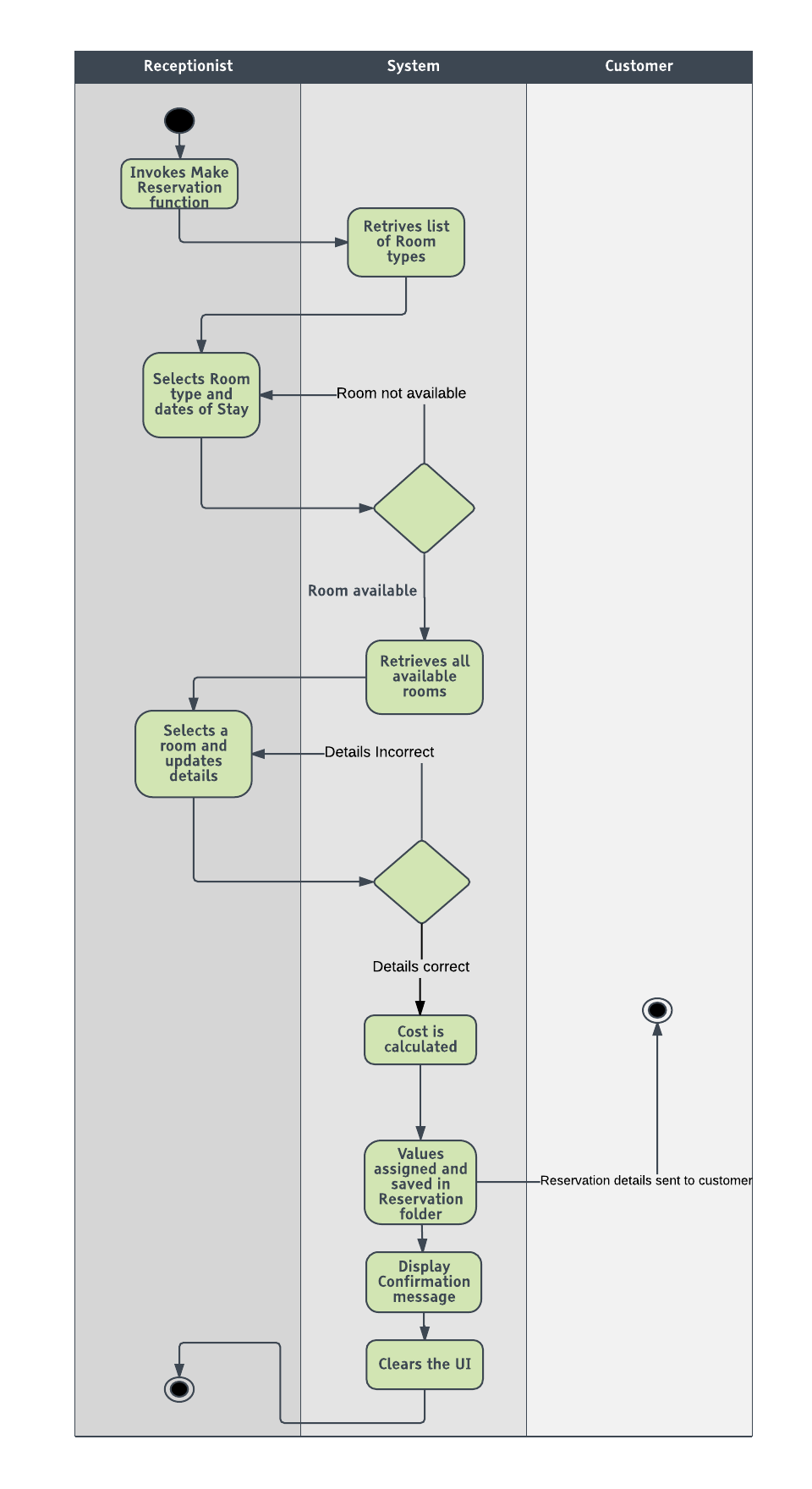
Customer

Receptionist

<<includes>>

«includes»

<<extends>>



|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Make Reservation** | |
| **Use Case Id** | 4.3.1 | |
| **Priority** | High | |
| **Source** | Manager | |
| **Primary Business Actor** | Receptionist | |
| **Other Participating Actors** | Customer | |
| **Description** | The purpose of this function is to allow the user to make a reservation | |
| **Preconditions** | The room must be available for a customer to book it. | |
| **Trigger** | N/A | |
| **Expected Scenario** | **Receptionist** | **System Response** |
|  | **Step 1: The Receptionist invokes the Make Reservation Function.**  **Step 4: The Receptionist then selects the Room\_Type from a list:**   * **Room\_Type**   **The Receptionist then updates specific fields:**   * **Date\_Arrive** * **Date\_Depart**   **Then selects the search for Reservation button**  **Step 10: The Receptionist then select the room no from a list**   * **Room\_No**   **The Receptionist then updates specific fields in relation to the booking**   * **Cust\_Name** * **E-mail**   **If the Customer is happy with the reservation details, the receptionist selects the confirm button** | **Step 2: System validates that there is already a Roomrate created before allowing you to update**   * **Roomrate not in existence**   **Step 3: The system retrieves a list of Room\_Type from the rooms file and displays the UI.**  **Step 5: The system then validates the search availability details:**   * **All fields must be entered** * **Date\_Depart can’t be a date before Date\_Arrive** * **Rooms unavailable**   **Step 6: The System retrieves all available rooms that match the Room\_Type and are not booked and are not closed (Status = ‘C’) for the nights in question from the rooms file and loads them on to the UI.**  **Step 7: The System retrieves the Rate from the Room Rates File and the type From the Rooms File**  **Step 8: The System generates a Cost (price) based on the Date\_Arrive/ Date\_Depart dates and Room\_Type(Days in between Date\_Arrive/Date\_Depart multiplied by the Room\_Type rate). This value is shown in the Total Cost textbox before the customer confirms the reservation**  **Step 9: Res\_ID is assigned a value**  **Step 11: The System validates the Reservation details:**   * **All field must be entered** * **Field only accepts Letters**     **Step 12: Res\_Status is assigned a value “A”**  **Step 13: System sends an E-mail to the customer confirming the reservation**  **Step 14: Save Reservation details in the Reservations file:**   * **Res\_ID** * **Room\_No** * **Cust\_Name** * **E-mail** * **Date Arrive** * **Date Depart** * **Cost** * **Res\_Status**   **Step 15: Display Confirmation Message**  **Step 16: Clear The UI** |
| **Alternate Scenarios** | **Receptionist** | **System Response** |
| **RoomRate not in existence** | **Step 1: Manager invokes the make Reservation function** | **Step 2: System finds that no Room Type has been entered into the system yet and displays the error message “There are no rooms available for booking”**  **Step 3: System clears the UI and return to Step 1** |
| **Field not entered** |  | **Step 5: A blank field entered**  **Step 6: Display message “This Field must be entered”**  **Step 7: Position cursor in offending field and return to step 4** |
| **Date Depart before Date Arrive** |  | **Step 5: Date Depart is before Date Arrive**  **Step 6: Display error message “Sorry Date Depart has to be a date after date arrive”**  **Step 7: Position cursor in the offending field and return to step 4** |
| **No Room available** |  | **Step 5: No room is available for the dates in question matching that Room\_Type.**  **Step 6: Display error message “Sorry there are no available rooms for the period selected please select another date/type”**  **Step 7: Position cursor in Room\_Type field and return to step 4** |
| **Field only accepts letters** |  | **Step 11: A Numeric Value is entered into Cust\_Name**  **Step 12: Display Message “This textbox only accepts letters”**  **Step 13: Position cursor in the offending field and return to step 10** |
| **Field not entered** |  | **Step 11: A blank field entered**  **Step 12: Display message “This Field must be entered”**  **Step 13: Position cursor in offending field and return to step 10** |
| **Customer not happy with some aspect of the reservation** | **Step 10: Receptionist selects the redo button** | **Step 11: Ui resets the total cost and returns the user to the original UI**  **Step 12: Receptionist repeats the reservation process at step 4** |
| **Conclusions** | Reservation is successfully made | |
| **Post conditions** | N/A | |
| **Business Rules** | A reservation can only made on pre-existing rooms  A rate must have been made in order to book a room | |
| **Implementation Constraints** | A reservation cannot be made on a closed room | |

### Cancel Reservation

HotelReservation\_SYS will allow a user to cancel a reservation

Customer

Receptionist

«includes»

<<extends>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Cancel Reservation** | |
| **Use Case Id** | 4.3.2 | |
| **Priority** | High | |
| **Source** | Manager | |
| **Primary Business Actor** | Receptionist | |
| **Other Participating Actors** | Customer | |
| **Description** | The purpose of this feature is to allow the user to cancel a reservation | |
| **Preconditions** | The room must be booked for a receptionist to cancel it. | |
| **Trigger** | N/A | |
| **Expected Scenario** | **Receptionist** | **System Response** |
|  | **Step 1: The Receptionist invokes the Cancel Reservation Function.**  **Step 4: The Receptionist then**  **selects the reservation she wants to delete based on:**   * **Res\_ID** * **Cust\_Name** | **Step 2: System validates that there is already Reservation in the system available for deletion on the criteria that the reservation has been made and not checked-in or out**   * **No available Reservation for deletion**   **Step 3: The system retrieves all previously made reservations from the Reservations file and displays the UI.**  **Step 5: The system then populates the Customer name textbox, Date arrive textbox, Date depart textbox and Res\_ID textbox (this is done so that the receptionist is certain they are deleting the correct reservation)**  **Step 6: The reservation is deleted from the system**  **Step 7: System send an E-mail to the customer confirming cancelation**  **Step 8: Display Confirmation Message**  **Step 9: Clear The UI** |
| **Alternate Scenarios** | **Receptionist** | **System Response** |
| **No available Reservation for deletion** | **Step 1: Manager invokes the Cancel Reservation function** | **Step 2: System finds that no Reservation is available for deletion and displays the message “There are no reservations available to be cancelled”**  **Step 3: System clears the UI and return to Step 1** |
| **Conclusions** | Reservation is successfully cancelled | |
| **Post conditions** | N/A | |
| **Business Rules** | No refund as the transaction doesn’t actually take place till the check\_out function  A Reservation that has checked In cannot be deleted | |
| **Implementation Constraints** | N/A | |

### 4.4.3. Check In

HotelReservation\_SYS will allow a user to check in

Customer

Receptionist

«includes»

<<extends>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Check In** | |
| **Use Case Id** | 4.3.3 | |
| **Priority** | Medium | |
| **Source** | Manager | |
| **Primary Business Actor** | Receptionist | |
| **Other Participating Actors** | Customer | |
| **Description** | The purpose of this function is to allow the user to allow a customer to Check\_In | |
| **Preconditions** | The customer must have already made a reservation | |
| **Trigger** | N/A | |
| **Expected Scenario** | **Receptionist** | **System Response** |
|  | **Step 1: The Receptionist invokes the Check\_In Function.**  **Step 4: The Receptionist then**  **selects the reservation she wants to check in based on:**   * **Res\_ID** * **Cust\_Name**   **Step 6: The Receptionist then selects the check\_In radio button**   * **Check\_In**   **And then selects the confirm button** | **Step 2: The system validates that there is a reservation available for check\_In on the criteria that a reservation has been made and the reservation is for today**   * **No available Reservation for check\_In**   **Step 3: The system displays the UI.**  **Step 5: The system then populates the Customer name textbox, Date depart textbox and Res\_Id textbox (this is done so that the receptionist is certain that they are checking in the correct reservation)**  **Step 7: The system validates the Check In details:**   * **Field not entered**   **Step 8: Check In Details Updated in the Reservations file**   * **Check\_In** * **Res\_ID**   **Step 9: Display Confirmation Message**  **Step 10: Clear The UI** |
| **Alternate Scenarios** | **Receptionist** | **System Response** |
| **No available Reservation for Check\_In** | **Step 1: Manager invokes the Check In function** | **Step 2: System finds that no Reservation is available for Check\_In and displays the message “There are no available reservations for check\_In today”**  **Step 3: System clears the UI and return to Step 1** |
| **Field not entered** |  | **Step 7: A blank field entered**  **Step 8: Display message “This Field must be entered”**  **Step 9: Position cursor in offending field and return to step 6** |
| **Conclusions** | Check In is successfully registered | |
| **Post conditions** | N/A | |
| **Business Rules** | In order for check-In to be made the reservation must have been for that day | |
| **Implementation Constraints** | N/A | |

### 4.4.4 Check Out

HotelReservation\_SYS will allow a user to check out.

Customer

Receptionist

< <<includes>>

«includes»

<<extends>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Check Out** | |
| **Use Case Id** | 4.3.4 | |
| **Priority** | Medium | |
| **Source** | Manager | |
| **Primary Business Actor** | Receptionist | |
| **Other Participating Actors** | Customer | |
| **Description** | The purpose of this feature is to allow the user to allow a customer to check Out and for the hotel to confirm payment | |
| **Preconditions** | The customer must have already Checked In | |
| **Trigger** | N/A | |
| **Expected Scenario** | **Receptionist** | **System Response** |
|  | **Step 1: The Receptionist invokes the Check\_Out Function.**  **Step 4: The Receptionist then**  **selects the reservation she wants to check out based on:**   * **Res\_ID** * **Cust\_Name**   **Step 6: The Receptionist then collects the money from the guest based on the agreed upon cost. The Receptionist then selects the check\_Out radio button**   * **Check\_Out**   **And then selects the confirm button** | **Step 2: The system validates that there is a reservation available for check\_Out on the criteria that a reservation is active and that said reservation has already been checked in**   * **No available Reservation for check\_Out**   **Step 3: The system displays the valid Reservations for check\_Out.**  **Step 5: The system then populates the Customer name textbox, Date Arrive textbox and Res\_Id textbox (this is done so that the receptionist is certain that they are checking Out the correct reservation)**  **Step 7: The system validates the Check Out details:**   * **Field not entered**   **Step 8: Res\_Status is assigned a value of “P”**  **Step 9: Check Out details updated in the Reservation file**   * **Check\_Out** * **Res\_Status**   **Step 10: Display confirmation message**  **Step11: Clear The UI** |
| **Alternate Scenarios** | **Receptionist** | **System Response** |
| **No available Reservation for Check\_Out** | **Step 1: Manager invokes the Check In function** | **Step 2: System finds that no Reservation is available for Check\_Out and displays the message “There are no available reservations for check\_Out today”**  **Step 3: System clears the UI and return to Step 1** |
| **Field not entered** |  | **Step 7: A blank field entered**  **Step 8: Display message “This Field must be entered”**  **Step 9: Position cursor in offending field and return to step 6** |
| **Conclusions** | Check Out is successfully registered | |
| **Post conditions** | N/A | |
| **Business Rules** | A Check Out can occur on any date after a guest has been check\_In however this those not entitle the guest to a refund | |
| **Implementation Constraints** | A Guest must have checked in, in order to check out | |

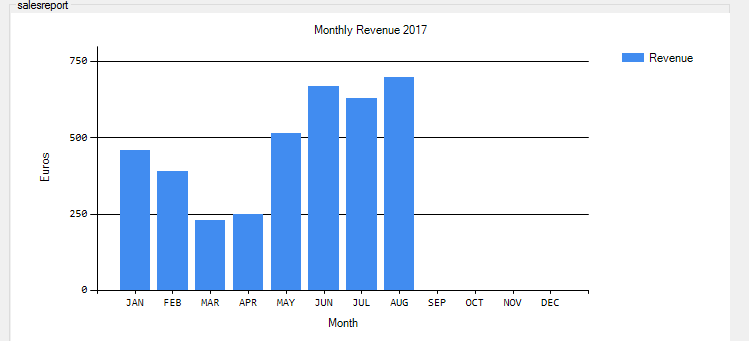
## Perform Admin

### Generate Sales Analysis

HotelReservation\_Sys will generate a sales report for any given year showing all monthly sales

Manager

|  |  |  |
| --- | --- | --- |
| **Use Case Id** | 4.4.1 | |
| **Priority** | Medium | |
| **Source** | Manager | |
| **Primary Business Actor** | Manager | |
| **Other Participating Actors** | N/A | |
| **Description** | The purpose of this function is to allow the user to generate a sales report | |
| **Preconditions** | N/A | |
| **Trigger** | N/A | |
| **Expected Scenario** | **Manager** | **System Response** |
|  | **Step 1: The admin invokes the Perform Sales Analysis function**  **Step 4: The user selects the year from a list to generate the sales report from** | **Step 2: The System displays the UI and retrieves relevant details from the Reservation file**  **Step 3: The system preloads the list box with relevant years (so the year in which a payment has been processed)**  **Step 5: The System generates Total\_Sales for the dates shown**  **(The sum of all sales for the given month(were Res Status is equal to “P”))**  **Step 6: The System generates the graph for the given year**  **Step 7: Displays the graph and a confirmation message**  **Step 8: manager is given the option of printing the graph**  **Step 9:** **Clear the UI** |
| **Alternate Scenarios** | **Manager** | **System Response** |
| **Conclusions** | A Sales Report has successfully been generated | |
| **Post conditions** | N/A | |
| **Business Rules** | N/A | |
| **Implementation Constraints** | N/A | |

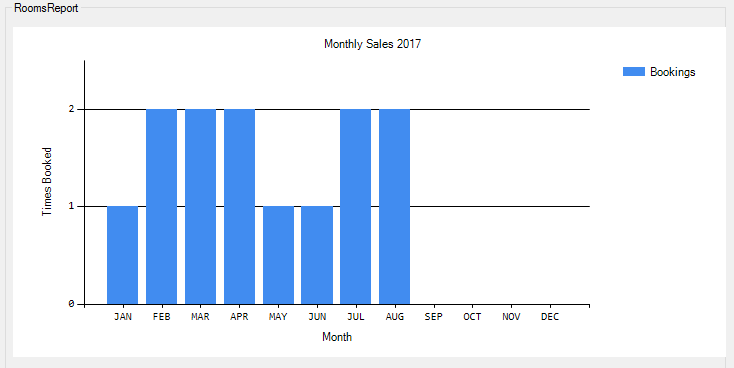


### Generate Rooms Analysis

HotelReservation\_SYS will generate a detailed Room Analysis report

Manager

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Generate Rooms Analysis** | |
| **Use Case Id** | 4.4.2 | |
| **Priority** | Medium | |
| **Source** | Manager | |
| **Primary Business Actor** | Manager | |
| **Other Participating Actors** | N/A | |
| **Description** | The purpose of this function is to allow the user to generate a Rooms report | |
| **Preconditions** | N/A | |
| **Trigger** | N/A | |
| **Expected Scenario** | **Manager** | **System Response** |
|  | **Step 1: The admin invokes the Generate Rooms Analysis function**  **Step 4: The user selects what year and Room\_type he would like to generate the report for** | **Step 2: The System displays the UI and retrieves relevant details from the Reservations**  **Step 3: The system preloads the year list box with relevant years (so the year in which a payment has been processed), and also preloads the Room Type list box with the relevant types(so types of rooms that have been booked in a given year)**  **Step 5: The system displays a chart based on the amount of times a room type was booked for a given year**   * **Room\_Type** * **Year**   **Step 7: Displays the graph and a confirmation message**  **Step 8: Allows manager to print the document if he wishes**  **Step 9**: **Clear the UI** |
| **Alternate Scenarios** | **Manager** | **System Response** |
| **Conclusions** | A Rooms Report has successfully been generated | |
| **Post conditions** | N/A | |
| **Business Rules** | N/A | |
| **Implementation Constraints** | N/A | |



# System Model

The following dataflow diagrams have been produced for the system:

## Level-0 DFD

Reservation request

Guest

HotelResevation\_SYS

Reservation confirmation

## Level-1 DFD

Guest

Room Rate Details

P1

Manage

Rooms

Confirmation E-Mail

Reservation Details

Rate Details

D1

Room Rate File

P2

Manage Reservations

Room\_Type Details

Room Details

D2

Rooms File

Room details

Reservation Details

D3

Reservations file

P3

Perform

Admin

Reservation Details

## Level-2 DFD (Manage Rooms)

P1.2

Update Room Rate

P1.1

Add Room Rate

Updated Room Rate Details

Room rate Details

Current Room Rate Details

P1.3

Add Room

D1

Room Rate file

Room Rate Details

Room Details

D2

Rooms file

Updated Room Details

Room Details

Room Status

Updated Room Status

Room Rate Details

P1.4

Update Room

P1.5

Close/Open Room

D1

Room Rate file

## Level-2 DFD (Manage Reservations)

Guest

Request Reservation

Cancel Reservation

Confirm Reservation

P2.2

Cancel Reservation

P2.1

Make

Reservation

Delete Reservation

Reservation Details

Reservation details

Check In Status

Room\_Type Details

D3

Reservations file

Rate Details

P2.3

Check

In

Reservation Details

Check Out status/Res status paid

Reservation Details

D1

Room Rate file

P2.4

Check

Out

Room Status

Room Status

D2

Rooms file

## Level-2 DFD (Perform Admin)

Payment details and dates

Pricing information

Room Details

D2

Rooms file

D3

Reservations file

P3.2

Generate Rooms Analysis

P3.1

Generate Sales Analysis

# Data Model (Class Diagram)

Brief introduction……

## Class Diagram



## Relational Schema

RoomRate{ Room\_Type, Description, Rate}

Rooms{ Room\_No, Room\_Type, Status}

Reservations{Res\_ID, Room\_No, Cust\_Name, E\_Mail, Date\_Arrive, Date\_Depart, Cost, Check\_In , Check\_Out, Res\_Status}

## Database Schema

**Relation:** Room Rate

**Attributes:**

Room\_Type char(3) NOT NULL UNIQUE

Description varchar2(30) NOT NULL

Rate decimal (5,2) NOT NULL

**Primary Key:** Room\_Type

**Relation:** Rooms

**Attributes:**

Room\_No numeric(3) NOT NULL UNIQUE

Room\_Type char(3)NOT NULL

Status char(1) NOT NULL

**Primary Key:** Room\_No

**[Foreign Key:** Room\_Type REFERENCES RoomRate (Room\_Type)]

**Relation:** Reservations

**Attributes:**

Res\_ID numeric(6) NOT NULL AUTO\_INCREMENT

Room\_No numeric(3) NOT NULL

Cust\_Name char(30) NOT NULL

E\_Mail varchar(30) NOT NULL

Date\_Arrive date NOT NULL

Date\_Depart date NOT NULL

Total\_Cost decimal (5,2) NOT NULL

Check\_In char (1) DEFAULT ‘A’ NOT NULL

Check\_Out enum(‘N’, ‘Y’) DEFAULT ‘A’ NOT NULL

Res\_Status char(1) DEFAULT ‘A’ NOT NULL

**Primary Key:** Res\_ID

**[Foreign Key:** Room\_No REFERENCES Rooms (Room\_No)]

# Conclusion

The aim of HotelReservation\_SYS was to provide a fully functional specification that would be relevant to the industry. This document contains specific user and system requirements which describe in detail the functions that said system must perform. Diagrams as well as detailed user case narratives were used to show the interaction between each process and the database and how these interactions were carried out and performed. This document also contained detailed information of how a user would interact with the system

I used Visual studio to create a visual representation of how the software should look and behave. One of the most important aspects I placed on this prototype was ensuring that it was user friendly and aesthetically pleasing

I believe that HotelReservation\_SYS contains all the necessary functionality that would be required by any hotel in the industry, however having said that I also see that this system has an excellent opportunity to expand its scope. For example, additional features such as adding multiple hotels to the system, forecasting expected profits based on previous years etc would really allow this product to grow.

# Appendices

## Appendix A – Title

## Appendix B – Title