Requirements Engineering

ReserveSys

Submitted By: David Herlihy T00196758

Computing with MultiMedia

Date Submitted: 12/12/2017

**Table of Contents**

[1. Introduction/overview 4](#_Toc498350120)

[2. Functional Components 5](#_Toc498350121)

[3. User Requirements 6](#_Toc498350122)

[4. System Requirements 7](#_Toc498350123)

[4.1. Process Room 8](#_Toc498350124)

[**4.1.1.** **Add Room Rate** 8](#_Toc498350125)

[**4.1.2.** **Update Rate** 10](#_Toc498350126)

[**4.1.3.** **Add Room** 13](#_Toc498350127)

[**4.1.4.** **Close Room** 17](#_Toc498350128)

[**4.1.5.** **Update Room** 19](#_Toc498350129)

[4.2. Process Reservation 21](#_Toc498350130)

[**4.2.1.** **Make Reservation** 21](#_Toc498350131)

[**4.2.2.** **Cancel Reservation** 25](#_Toc498350132)

[**4.2.3.** **Process No-Shows** 27](#_Toc498350133)

[**4.2.4.** **Process Check-in** 30](#_Toc498350134)

[**4.2.5.** **Process Check-Out** 33](#_Toc498350135)

[4.3. Analysis 36](#_Toc498350136)

[**4.3.1.** **Room Sales Analysis** 36](#_Toc498350137)

[**4.3.2.** **Monthly Sales Analysis** 39](#_Toc498350138)

[5. System Model 42](#_Toc498350139)

[5.1. Level-0 DFD 42](#_Toc498350140)

[5.2. Level-1 DFD 43](#_Toc498350141)

[5.3. Level-2 DFD (Process P1: Process Rooms) 44](#_Toc498350142)

[5.4. Level-2 DFD (Process P2: Process Reservations) 45](#_Toc498350143)

[5.5. Level-2 DFD (Process P3: Generate Analysis) 46](#_Toc498350144)

[6. Data Model (Class Diagram) 47](#_Toc498350145)

[6.1. Class Diagram 47](#_Toc498350146)

[6.2. Relational Schema 47](#_Toc498350147)

[6.3. Database Schema 47](#_Toc498350148)

[7. Conclusion 48](#_Toc498350149)

[8. Appendices 49](#_Toc498350150)

[8.1. Appendix A – Title 49](#_Toc498350151)

[8.2. Appendix B – Title 49](#_Toc498350152)

# Introduction/overview

The aim of this project is to plan out a system for a Hotel. This system, called ReserveSys, will be able to perform services for the Hotel to process Rooms and Reservations. ReserveSys will also be able to provide administration functions such as sales and occupation reports for The Hotel.

Within the Rooms function, ReserveSys will allow an administrator to manage the Rates and Rooms of the Hotel. This will allow the admin control of the rooms in the hotel, being able to add new rooms, update the details of existing a room and close a room. The system also allows an admin to add a rate to the system which can be assigned to a room and update the details of a current rate.

Within the Reservations function, ReserveSys allows a member of staff to handle the Reservations of the Hotel. A staff member will be able to create and cancel Reservations. Upon arrival of a guest the staff member will be able to check in and check out the guest. If a guest does not show up on the date set in the reservation the staff member will be able to process the no-show and charge the guest.

In terms of Administration, ReserveSys will provide functions to generate two reports. The Room Type Analysis will generate a report based on a Room Type and a year both set by the administrator, showing how many bookings, the hotel had during the year divided into months for specific analysis. This allows the administrator to gauge what is the most popular period for a Room Type. The Yearly Sales analysis generates a sales reports for a year, calculating the Monthly sales of the hotel for each month in the year specified. This allows the user to calculate when is the busiest period overall for the hotel.

The aim of ReserveSys is to provide the user accessible daily management of the Hotel as well as analysis for the Hotel to allow the managers to analyse the performance of the hotel and its different room types over a yearly period and prepare for the future.

# Functional Components

# User Requirements

1. ReserveSys will perform Room administration.

1.1. ReserveSys will add a rate to the system to be assigned to a room.

1.2. ReserveSys will update the rates in the system.

1.3. ReserveSys will add a new room.

1.4. ReserveSys will close a current room.

1.5. ReserveSys will update the details of a room.

2. ReserveSys will perform Reservation administration.

2.1. ReserveSys will process a reservation.

2.2. ReserveSys will process a cancellation of a reservation.

2.3. ReserveSys will process a check in and charge a customer

2.4. ReserveSys will process a check out.

2.5. ReserveSys will process a no-show and charge a customer

3. ReserveSys will provide analysis

3.1. ReserveSys will generate a report based on Room Type and a year.

3.2. ReserveSys will generate a sales report based on Monthly Sales for a year.

# System Requirements

**System Level Use Case Diagram**

ReserveSys

Staff

Admin

Customer

## 4.1. Process Rooms

The Rooms function will allow the user to manage the Rates and Rooms in the system. The function allows the user to add and update rates and add, close and update rooms.

### **Add Room Rate**

* This function adds a new rate for rooms to the Rates File. This rate can now be assigned to rooms.

Admin

<<includes>>

<<extends>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Add Room rate** | |
| **Use Case Id** | 4.1.1. | |
| **Priority** | Medium | |
| **Source** | Manager | |
| **Primary Business Actor** | Admin | |
| **Other Participating Actors** |  | |
| **Description** | This function adds a new rate for rooms to the Rates File | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Admin** | **System** |
|  | **Step 1:** The admin invokes the Add Rate function  **Step 3:** The Admin enters the details for the room rate:   * Room\_Type * Rate\_Description * Room\_Rate  |  | | --- | |  | | **Step 2:** The System displays the UI  **Step 4:** The System validates the room details:   * All fields must be entered * Description must not contain numeric values * Rate must only contain numeric values   **Step 5:** Save the details to the Rates File   * Room\_Type * Rate\_Description * Room\_Rate   **Step 6:** Display confirmation message  **Step 7:** Clear the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Field not entered** |  | **Step 3:** A blank field has been detected  **Step 4:** Display error message “This field must be entered/selected”  **Step 5:** Position cursor in the offending field and return to step 3 |
| **Numeric data detected** |  | **Step 3:** A numeric value has been entered in a text field  **Step 4:** Display error message “This field must not be numeric”  **Step 5:** Position cursor in the offending field and return to step 3 |
| **Non-Numeric data detected** |  | **Step 3:** A non-numeric value has been entered in a text field  **Step 4:** Display error message “This field must be numeric”  **Step 5:** Position cursor in the offending field and return to step 3 |
| **Conclusions** | A new room rate has been added to the system | |
| **Post conditions** | This rate can now be assigned to a Room | |
| **Business Rules** |  | |

### **Update Rate**

* This function allows the Admin to update a Rate in the Rates File.

<<includes>>

<<extends>>

Admin

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Update Rate** | |
| **Use Case Id** | 4.1.2. | |
| **Priority** | Medium | |
| **Source** | Manager | |
| **Primary Business Actor** | Admin | |
| **Other Participating Actors** |  | |
| **Description** | This function updates a rate for rooms in the Rates File | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Admin** | **System** |
|  | **Step 1:** The admin invokes the Update Rate function  **Step 3:** The Admin selects the Rate that is to be updated   |  | | --- | |  |   **Step 5:**  The Admin updates the details for the rate:   * Description * Room\_Rate | **Step 2:** The System retrieves Rates from the Rates File  and displays the UI  **Step 4:** The system retrieves the details for the rate from the Rates File and displays the UI  **Step 6:** The System validates the rate details:   * All fields must be entered * Description must not contain numeric values only * Room\_Rate must only contain numeric values   **Step 7:** Update the details to the Rates File:   * Description * Room\_Rate   **Step 8:** Display confirmation message  **Step 9:** Clear the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Field not entered** |  | **Step 6:** A blank field has been detected  **Step 7:** Display error message “This field must be entered/selected”  **Step 8:** Position cursor in the offending field and return to step 5 |
| **Numeric data detected** |  | **Step 6:** A numeric value has been entered in a text field  **Step 7:** Display error message “This field must not be numeric”  **Step 8:** Position cursor in the offending field and return to step 5 |
| **Non-Numeric data detected** |  | **Step 6:** A non-numeric value has been entered in a text field  **Step 7:** Display error message “This field must be numeric”  **Step 8:** Position cursor in the offending field and return to step 5 |
| **Conclusions** | A room rate has been updated in the system | |
| **Post conditions** | This rate can now be assigned to a Room | |
| **Business Rules** |  | |

### **Add Room**

* The Add Room function will allow the admin to add rooms to the Rooms File.

When a room is added its status is set to available, this status may later be set to unavailable when a guest checks into the room. When the room is closed the status is set to closed.

<<includes>>

<<extends>>

Admin

|  |  |
| --- | --- |
| Activity Diagram: Add Room | |
| Admin | System |
| Enter Room Details  Invoke Add Room Function | Set Status to Available (‘A’)  Save Room Details  Display Confirmation Message  Reset UI  No  Yes  Valid  Display UI  Error Message  Validate Room Details |

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Add Room** | |
| **Use Case Id** | 4.1.3. | |
| **Priority** | Medium | |
| **Source** | Manager | |
| **Primary Business Actor** | Admin | |
| **Other Participating Actors** |  | |
| **Description** | This function adds rooms to the Rooms File | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Admin** | **System** |
|  | **Step 1:** The admin invokes the Add Room function  **Step 3:** The Admin selects the details for the room:   * Room\_Type * Room\_Description  |  | | --- | |  | | **Step 2:** The System retrieves Room\_Type from the Rates File, generates a new Room No then loads the UI  **Step 4:** The System validates the room details:   * Room\_Type/ Description must be selected/entered   **Step 5:** Set Room\_Status to ‘A’ for available.  **Step 6:** Save the Room details in the Rooms File:   * RoomNo * Room\_Description * Room\_Type * Room\_Status   **Step 7:** Display confirmation message  **Step 8:** Clear the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Field not entered/selected** |  | **Step 4:** The system has detected a field that has not been entered or selected.  **Step 5:** Display error message “This item must be selected”  **Step 6:** Position cursor on the offending field and return to step 3 |
| **Conclusions** | A new room is registered on the system | |
| **Post conditions** | This room can now be reserved. | |
| **Business Rules** |  | |

### **Close Room**

* The Close Room function allows the user to Close a room. When a room is closed its status is set to “C” for closed.

<<includes>>

<<extends>>

Admin

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Close Room** | |
| **Use Case Id** | 4.1.4. | |
| **Priority** | Medium | |
| **Source** | Manager | |
| **Primary Business Actor** | Admin | |
| **Other Participating Actors** |  | |
| **Description** | This function will close a room, settings the rooms status to “C” for closed. | |
| **Preconditions** | A room must not have any reservations at the time of or after the time of closing the room in order to be closed | |
| **Trigger** |  | |
| **Expected Scenario** | **Admin** | **System** |
|  | **Step 1:** The admin invokes the Close Room function  **Step 4:** The Admin selects the room that is to be removed from the list   * RoomNo  |  | | --- | |  |   **Step 7:** The Admin confirms that this room is to be removed from the system   * Yes/No | **Step 2:** The system retrieves all unoccupied rooms which have no current or future reservations from the Room File and the Reservations File  **Step 3:** The System displays the UI  **Step 5:** The System validates the room details:   * RoomNo must be selected     **Step 6:** The System displays the rooms details and asks for conformation   * RoomNo * Room\_Type   **Step 8:** The system verifies if the user selected Yes or No  **Step 9:** Change the Room status to “C” for closed in the Rooms File  **Step 10:** Display confirmation message  **Step 11:** Clear the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Field not entered/selected** |  | **Step 5:** The system has detected a field that has not been entered or selected.  **Step 6:** Display error message “This item must be selected”  **Step 7:** Position cursor on the offending field and return to step 3 |
| **The room is unavailable** |  | **Step 4:** There is a reservation in the Reservations File which the Sys\_Date is between the Arrival\_Date and Dept\_Date  **Step 5:** Display error message “This room is currently in use and cannot be removed from the system”  **Step 6:** Position cursor on the Room selection and return to step 3 |
| **User Selected “No”** |  | **Step 8:** The system detected a NO answer  **Step 9:** Return to step 4 |
| **Conclusions** | A room is removed from the system | |
| **Post conditions** | This room can no longer be reserved. | |
| **Business Rules** | A room cannot be removed if it is in use on that day | |

### **Update Room**

* This function allows the Admin to update a Room’s details. A room’s details can be changed if it has reservations either at the time of updating or upcoming as the cost is calculated when the reservation is booked.

<<includes>>

<<extends>>

Admin

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Update Room** | |
| **Use Case Id** | 4.1.5. | |
| **Priority** | Medium | |
| **Source** | Manager | |
| **Primary Business Actor** | Admin | |
| **Other Participating Actors** |  | |
| **Description** | This function updates a Room’s details in the Rooms File | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Admin** | **System** |
|  | **Step 1:** The admin invokes the Update Room function  **Step 4:** The Admin selects the Room that is to be updated   * RoomNo  |  | | --- | |  |   **Step 8:**  The Admin updates the details for the rate:   * Room\_Type * Description | **Step 2:** The System retrieves a list of Rooms from the Rooms File and shows the rooms:   * RoomNo   **Step 3:** The System displays the UI  **Step 5:** The system performs validation:   * A RoomNo must have been selected   **Step 6:** The system retrieves the rooms details from the Rooms File  **Step 7:** The System displays the UI  **Step 9:** The System validates the room details:   * Room\_Type/ Description must be selected/entered   **Step 10:** If a room’s status is set to ‘C’ for closed then its status is set to ‘A’ for available. This is how the Admin reopens a room.  **Step 11:** Save the details to the Rooms File   * RoomNo * Room\_Type * Description * Room\_Status(Does not change)   **Step 12:** Display confirmation message  **Step 13:** Clear the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Field not selected** |  | **Step 5:** A blank field has been detected  **Step 6:** Display error message “This field must be entered/selected”  **Step 7:** Position cursor in the offending field and return to step 4 |
| **Field not selected/entered** |  | **Step 8:** A blank field has been detected  **Step 9:** Display error message “This field must be entered/selected”  **Step 10:** Position cursor in the offending field and return to step 8 |
| **Conclusions** | A room has been updated in the system | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

## Process Reservation

The Process Reservation function allows the user to process reservation, check ins and outs and no-shows.

### **Make Reservation**

* The Make Reservation function allows the user to add a reservation to the Reservation File. Only one room can be reserved per reservation.

Customer

Staff

|  |  |
| --- | --- |
| Activity Diagram: Make Reservation | |
| Staff | System |
| Select Room  Enter Reservation Details  Invoke Make Reservation Function | Validate Room Details  Display Customer UI  Enter Customer Details  Validate Room Details  Error Message  No  Yes  Valid  Display Room UI  Error Message  No  Yes  Valid  Validate Reservation Details  Display Reservation UI |

|  |  |
| --- | --- |
|  | Display Confirmation Message  Save Reservations Details  Set Reservation Status to “B”  Save Customer Details  Error Message  No  Yes  Valid |

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Make Reservation** | |
| **Use Case Id** | 4.2.1. | |
| **Priority** | High | |
| **Source** | Manager | |
| **Primary Business Actor** | Staff | |
| **Other Participating Actors** | Customer | |
| **Description** | This function adds a reservation for a room to the Reservations File | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Staff** | **System** |
|  | **Step 1:** The user invokes the Make Reservation function  **Step 4:** The user enters the details for the reservation:   * Arrival\_Date * Dept\_Date * Room\_Type   **Step 9:** The user selects a Room:   * RoomNo   **Step 13:** The system enters the details for the customer:   * Name-Fname, Sname * Address- Street, Town, County * Tel\_No * Payment Details- CardNo,   CardHolder\_Name | **Step 2:** The System retrieves the last ResNo from the Reservations file and generates a new number and retrieves a list of Room\_Types from the rates file.  **Step 3:** The System displays the UI for entering a Reservation  **Step 5:** The system validates the details entered:   * Arrival\_Date/Dept\_Date   must be selected correctly   * All fields must be selected   **Step 6:** The system retrieves a list of rooms available during the selected period and of selected type from the Rooms file.  **Step 7:** The system validates that there are rooms available to be selected.  **Step 8:** The system displays a list of available rooms to be selected  **Step 10:**  The system performs validation:   * that a RoomNo has been selected   **Step 11:** The System retrieves the last CustID from the Customers file and generates a new ID  **Step 12:** The system displays the UI for registering a customer  **Step 14:** The System validates the details entered:   * Name/Address/ * CardHolder\_Name must be non-numeric * Tel\_No/CardNo must be numeric * All fields must be entered   **Step 15:** The system saves the details to the Customers File:   * CustID * Name-Fname, Sname * Address- Street, Town, County * Tel\_No * Payment Details-   CardNo, CardHolder\_Name  **Step 16:** The system sets the status of the Reservation to “B” for Booked  **Step 17:** The system calculates the Cost of the reservations by getting the period between the Arrival\_Date and Dept\_Date and multiplies it by the Room\_Rate from the Rates File.  **Step 18:** The system saves the details to the reservations file:   * ResNo * Arrival\_Date * Dept\_Date * Res\_Status * Cost * RoomNo * CustID   **Step 19:** The system displays a confirmation message  **Step 20:** Clear the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Numeric data detected** |  | **Step 13:** A numeric value has been entered in a text field  **Step 14:** Display error message “This field must not be numeric”  **Step 15:** Position cursor in the offending field and return to step 12 |
| **Non-Numeric data detected** |  | **Step 13:** A non-numeric value has been entered in a text field  **Step 14:** Display error message “This field must be numeric”  **Step 15:** Position cursor in the offending field and return to step 12 |
| **Field not entered** |  | **Step 13:** A blank field has been detected  **Step 14:** Display error message “This field must be entered”  **Step 15:** Position cursor in the offending field and return to step 12 |
| **Field not selected** |  | **Step 8:** A blank field has been detected  **Step 9:** Display error message “This field must be entered/selected”  **Step 10:** Position cursor in the offending field and return to step 7 |
| **No rooms available** |  | **Step 7:** There are no results from the search  **Step 8:** Display error message stating there are no rooms of this type available during the selected period  **Step 9:** Return to step 4. |
| **Conclusions** | A new reservation has been made | |
| **Post conditions** | This room can no longer be reserved during the period set. | |
| **Business Rules** | A room cannot be reserved if there is a booking set during the period requested by the customer. | |
| **Implementation Constraints** |  | |

### **Cancel Reservation**

* The Cancel reservation function allows the user to cancel a reservation on the system. After the reservation is cancelled its status is changed to cancelled and the rooms status will stay set to unoccupied.

Customer

Staff

<<includes>>

<<extends>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Cancel Reservation** | |
| **Use Case Id** | 4.2.2. | |
| **Priority** | High | |
| **Source** | Manager | |
| **Primary Business Actor** | Staff | |
| **Other Participating Actors** | Customer | |
| **Description** | This function sets the status of a reservation for a room in the Reservations File to cancelled | |
| **Preconditions** |  | |
| **Expected Scenario** | **Staff** | **System** |
|  | **Step 1:** The user invokes the Cancel Reservation function  **Step 3:** The user enters the name of a customer:   * Sname * Fname   **Step 6:** The user selects the reservation requested to be cancelled by the customer.  **Step 9:** The user confirms that this reservation is to be removed from the system   * Yes/No | **Step 2:** The System displays the UI  **Step 4:** The system validates that the Customer details has been entered correctly  **Step 5:** The system retrieves a list of reservations that have the selected Fname and Sname and displays them for selection  **Step 7:** The system performs validation:   * Checks if a reservation has been selected * Checks if the Sys\_Date is greater that 24hrs away from the Arrival Date   **Step 8:** The system displays a confirmation message asking if this reservation is correct  **Step 10:** The system validates whether the user selected Yes or no  **Step 11:** Change the Res\_Status to “C” for Cancelled from the Reservations File  **Step 12:** Display a confirmation message stating the reservation has been cancelled.  **Step 13:** Clear the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Field not selected** |  | **Step 7:** A blank field has been detected  **Step 8:** Display error message “This field must be entered/selected”  **Step 9:** Position cursor in the offending field and return to step 6 |
| **SysDate is not greater than 24 hours from the Arrival Date** |  | **Step 7:** The Sys\_Date is less than 24hrs to the Arrival Date  **Step 8:** The system displays the error message: “Cancellations must occur 24 hours before the Date of Arrival. Unable to process.”  **Step 9:** Exit the function. |
| **User Selected “No”** |  | **Step 8:** Return to Step 4 |
| **Conclusions** | A reservation has been cancelled. The room is now free to be booked during the set period. | |
| **Post conditions** | This reservation is set to Cancelled and the customer will not be charged | |
| **Business Rules** | A reservation cannot be cancelled if it is less than 24 hours from the Arrival\_Date | |
| **Implementation Constraints** |  | |

### **Process No-Shows**

* This function processes no-shows and taking into account the date of arrival and charges to the payment file

<<includes>>

<<extends>>

Staff

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Process No-Shows** | |
| **Use Case Id** | 4.2.3. | |
| **Priority** | High | |
| **Source** | Manager | |
| **Primary Business Actor** | Staff | |
| **Other Participating Actors** |  | |
| **Description** | This function processes a no-show and charges the customer. | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Staff** | **System** |
|  | **Step 1:** The user invokes the Process No-Show function  **Step 4:** The user selects the Reservation that is to be processed:   * ResNo   **Step 9:** The user confirms that this is the correct customer to be charged:   * Yes/No | **Step 2:** The System retrieves a list of Reservations with a status of Booked where the Sys\_Date is 24 hours after the Arrival Date:   * ResNo * Name(Fname, Sname from Customers File) * RoomNo   **Step 3:** The System displays the UI  **Step 5:**  The system validates that a reservation has been selected  **Step 6:** The system retrieves the Payment Details from the Customer File.  **Step 7:** The System retrieves the Cost from the Reservations File.  **Step 8:** The System displays a confirmation message that this is the correct customer to be charged and show the amount owed   * Name(Fname, Sname) * RoomNo * Cost   **Step 10:**  The system validates whether the user has selected yes or no.  **Step 11:** Change the Res\_Status to “N “ for No-Show in the Reservations File  **Step 12:** The system generates a new PayID based on the previous one in the Payments File and saves the details:   * PayID * Pay\_Date=Sys\_Date * Cost * CustID * ResNo   **Step 13:** Display a confirmation message stating the reservation has been processed and payment has been processed.  **Step 14:** Clear the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Field not selected** |  | **Step 5:** A blank field has been detected  **Step 6:** Display error message “This field must be entered/selected”  **Step 7:** Position cursor in the offending field and return to step 4 |
| **User Selected “No”** |  | **Step 8:** Return to Step 4 |
| **Conclusions** | A reservation has been processed and payment has been processed | |
| **Post conditions** |  | |
| **Business Rules** | A no-show can only be processed if over 24 hours has passed from the Arrival Date | |

### **Process Check-in**

* The Process Check-in function checks a customer into the hotel and charges the customer for the reservation. Once a customer is checked in the Room’s status is set to occupied.

Customer

Staff

<<includes>>

<<extends>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Process Check-In** | |
| **Use Case Id** | 4.2.4. | |
| **Priority** | High | |
| **Source** | Manager | |
| **Primary Business Actor** | Staff | |
| **Other Participating Actors** | Customer | |
| **Description** | This function processes a check-in to the hotel | |
| **Preconditions** |  | |
| **Trigger** |  | |
| **Expected Scenario** | **Staff** | **System** |
|  | **Step 1:** The admin invokes the Process Check-In function  **Step 4:** The user selects the Reservation that is to be processed:   * ResNo   **Step 7:** The user selects Yes/No | **Step 2:** The System retrieves a list of Reservations with a status of Booked and checks that the SysDate is the same date as the Arrival Date, showing:   * ResNo * Name(Fname, Sname) * RoomNo   **Step 3:** The System displays the UI  **Step 5:** The system validates that a ResNo has been selected.  **Step 6:** The system displays a message asking whether this is the correct customer.  **Step 8:**  The system validates whether the user has selected yes or no  **Step 9:** The system retrieves the payment details from the customer file  **Step 10:** The system retrieves the cost from the Reservations File  **Step 11:** The system generates a new PayID based on the previous one in the Payments File and saves the details:   * PayID * Pay\_Date * Amount\_Paid * CustID * ResNo   **Step 12:** Changes the status of the reservation to “I” for Checked-In.  **Step 13:** The system changes the status of the room to occupied.  **Step 14:** Display a confirmation message stating the reservation has been processed and the customer is now checked in.  **Step 14:** Clear the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Field not selected** |  | **Step 5:** A blank field has been detected  **Step 6:** Display error message “This field must be entered/selected”  **Step 7:** Position cursor in the offending field and return to step 4 |
| **User Selected “No”** |  | **Step 8:** Return to Step 4 |
| **Conclusions** | A reservation has been processed and the customer has been “Checked-In”. The status of the room of the reservation is changed to occupied. | |
| **Post conditions** | The status of the reservation has been changed to “Checked-In”. The Customer has been charged and a payment has been added to the payments file. | |
| **Business Rules** | A reservation can only be checked in on the Date of Arrival.  A payment must be taken on check in. | |
| **Implementation Constraints** |  | |

### **Process Check-Out**

* The Process Check-in function checks a customer out of the hotel.

Customer

Staff

<<includes>>

<<extends>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Process Check-Out** | |
| **Use Case Id** | 4.2.5. | |
| **Priority** | High | |
| **Source** | Manager | |
| **Primary Business Actor** | Staff | |
| **Other Participating Actors** | Customer | |
| **Description** | This function processes a check-out from the hotel | |
| **Preconditions** |  | |
| **Expected Scenario** | **Staff** | **System** |
|  | **Step 1:** The admin invokes the Process Check-Out function  **Step 4:** The user selects the Reservation that is to be processed:   * ResNo | **Step 2:** The System retrieves all Reservations from Reservations with a status of “Checked-In” and checks that the SysDate is the same date as the Departure Date showing:   * ResNo * Name (Fname, Sname from Customers File) * RoomNo   **Step 3:** The System displays the UI  **Step 5:**  The system validates that a ResNo has been selected.  **Step 6:** Changes the status of the reservation to “O” for Checked-Out  **Step 7:** The system changes the status of the room to unoccupied  **Step 12:** Display a confirmation message stating the reservation has been processed and the customer is now checked out.  **Step 13:** Clear the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Field not selected** |  | **Step 5:** A blank field has been detected  **Step 6:** Display error message “This field must be entered/selected”  **Step 7:** Position cursor in the offending field and return to step 4 |
| **Conclusions** | A reservation has been processed and the customer has been “Checked-Out”. The room’s status is set to unoccupied. | |
| **Post conditions** | The status of the reservation has been changed to “Checked-Out”. | |
| **Business Rules** | A reservation can only be checked out on the Date of Departure | |

## Analysis

The Analysis process generates sales reports for the hotel

### **Room Type Analysis**

* The Room Type Analysis function generates a report based on a room type and a year, calculating how many times the room was booked per month for the selected year

<<includes>>

<<extends>>

Admin

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Room Sales Analysis** | |
| **Use Case Id** | 4.3.1. | |
| **Priority** | Medium | |
| **Source** | Manager | |
| **Primary Business Actor** | Admin | |
| **Description** | This function generates a report based on a room type and a year | |
| **Expected Scenario** | **Admin** | **System** |
|  | **Step 1:** The admin invokes the Room Sales Analysis function  **Step 4:** The user selects the Room Type and the year that is to be processed:   * Room\_Type * Year | **Step 2:** The System retrieves a list of Room\_Types from the Rates File, showing:   * Room\_Type * Description   **Step 3:** The System displays the UI  **Step 5:** The system validates that the details have been selected  **Step 6:** The system generates a report for the Room\_Type, showing the amount of times that type of room was booked per month for that year  **Step 7:** Displays the report and a confirmation message  **Step 8:** Clear the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Field not selected** |  | **Step 5:** A blank field has been detected  **Step 6:** Display error message “This field must be entered/selected”  **Step 7:** Position cursor in the offending field and return to step 4 |
| **Conclusions** | A report has been generated for the user displaying the sales details of a type of room during a selected period | |
| **Post conditions** |  | |
| **Business Rules** |  | |

### **Yearly Sales Analysis**

* The Yearly Sales Analysis function generates a sales report based on the sales of rooms per month in a year selected by the user

Admin

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Yearly Sales Analysis** | |
| **Use Case Id** | 4.3.1. | |
| **Priority** | Medium | |
| **Source** | Manager | |
| **Primary Business Actor** | Admin | |
| **Other Participating Actors** |  | |
| **Description** | This function generates a sales report based on the sales of the hotel in a year selected by the user | |
| **Preconditions** |  | |
| **Expected Scenario** | **Admin** | **System** |
|  | **Step 1:** The admin invokes the Yearly Sales Analysis function  **Step 3:** The user selects the Year that is to be analysed   * Year | **Step 2:**  The system displays the UI for selecting the period    **Step 4:** The system performs validation:   * All fields must be entered   **Step 5:** The system generates a report displaying the total sales per month for the year selected.  **Step 6:** Displays the report and a confirmation message  **Step 7:** Clear the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Field not entered** |  | **Step 4:** A blank field has been detected  **Step 5:** Display error message “This field must be entered/selected”  **Step 6:** Position cursor in the offending field and return to step 3 |
| **Conclusions** | A report has been generated on the sales of the hotel based on a year selected by the user | |
| **Post conditions** |  | |
| **Business Rules** |  | |

# System Model

The following dataflow diagrams have been produced for the system:

## Level-0 DFD

Customer Details

Customer

RESERVESYS

Receipt

## Level-1 DFD

Customer

Room Details

Room Details

Customer Details

P2

Process Reservations

D1

Rooms File

P1

Process Rooms

D2

Rates File

Rate Details

Customer Details

D3

Customers File

Reservation Details

D4

Reservations File

P3

Generate Analysis

Reservation Details

Payment Details

D5

Payments File

Payment Details

## Level-2 DFD (Process P1: Process Rooms)

P1.1

Add Room Rate

Rate Details

Updated Rate Details

P1.2

Update Room Rate

Rate Details

D2

Rates File

Room Details

P1.3

Add Room

Rate Details

Rate Details

P1.4

Update Room

Room Details

D1

Rooms File

Remove Room Details

Room Details

Updated Room Details

P1.5

Close Room

## Level-2 DFD (Process P2: Process Reservations)

Notification

Customer Details

Customer

Customer Details

P2.2

Cancel Reservation

P2.1

Make Reservation

D3

Customers File

Reservation Details

Reservation Details

D4

Reservations File

Reservation Details

Room Details

D1

Rooms File

P2.4

Process Check-In

Room Status

P2.3

Process No-Show

Payment Details

Notification

D5

Payments File

Payment Details

Payment Details

Room Status

D1

Rooms File

P2.5

Process Check-out

Room Status

Reservation Details

D4

Reservations File

Reservation Details

Notification

Customer

## Level-2 DFD (Process P3: Generate Analysis)

Payment Details

P3.2

Yearly Sales Analysis

Payment Details

Reservation Details

D5

Payments File

D4

Reservations File

P3.1

Room Type Analysis

# Data Model (Class Diagram)

In this data model there will be a class diagram showing how the files in the system interact with each other and showing the relational integrity.

## Class Diagram



## Relational Schema

Rates (Room\_Type, Rate\_Description, Room\_Rate)

Rooms (RoomNo, Room\_Description, Room\_Type, Room\_Status)

Customers (CustID, Fname, Sname, Street, Town, County, Tel\_No, CardNo, CardHolder\_Name)

Reservations (ResNo, Arrival\_Date, Dept\_Date, Res\_Status, Cost, CustID, RoomNo)

Payments (PayID, Pay\_Date, Amount\_Paid, CustID, ResNo)

## Database Schema

**Schema:** **ResSys Schema**

**Relation:** Rates

Attributes:

Room\_Type char (2) NOT NULL

Rate\_Description varchar2 (30) NOT NULL

Room\_Rate numeric (5, 2) NOT NULL

**Primary Key:** Room\_Type

**Relation:** Rooms

Attributes:

RoomNo numeric (6) NOT NULL

Room\_Description varchar2 (30) NOT NULL

Room\_Type char (2) NOT NULL

Room\_Status char (1) NOT NULL

**Primary Key:** RoomNo

**Foreign Key:** Room\_Type REFERENCES Room\_Type in the Rates table

**Relation:** Customers

Attributes:

CustID numeric (6) NOT NULL

Fname varchar2 (10) NOT NULL

Sname varchar2 (15) NOT NULL

Street1 varchar2 (25) NOT NULL

Town varchar2 (25) NOT NULL

County varchar2 (9) NOT NULL

Tel\_No varchar2 (15)

CardNo numeric (16) NOT NULL

CardHolders\_Name varchar2 (25) NOT NULL

**Primary Key:** CustID

**Relation:** Reservations

Attributes:

ResNo numeric (6) NOT NULL

Arrival\_Date date NOT NULL

Dept\_Date date NOT NULL

Res\_Status char (1) NOT NULL

CustID numeric (6) NOT NULL

RoomNo numeric (3) NOT NULL

**Primary Key:** ResNo

**Foreign Key:** CustID REFERENCES CustID in the Customers table

**Foreign Key:** RoomNo REFERENCES RoomNo in the Rooms table

**Relation:** Payments

Attributes:

PayID numeric(6) NOT NULL

Pay\_Date date NOT NULL

Amount\_Paid numeric (5, 2) NOT NULL

ResNo numeric (6) NOT NULL

CustID numeric (6) NOT NULL

**Primary Key:** PayID

**Foreign Key:** ResNo REFERENCES ResNo in the Reservations table

**Foreign Key:** CustID REFERENCES CustID in the Customer table

# Conclusion

After completing this report, ReserveSys will achieve all the objectives set beforehand. ReserveSys will successfully manage and store all the information need to run a hotel. By performing Room processes a manger will be able to add rooms to a Rooms File. Once added a rooms information will be able to be updated. A Room will have a status which will be changed from unoccupied to occupied as guests are checked in and out. A Room can be closed by changing this status to closed therefore retaining the relational integrity by not deleting any records. A room can be reopened by activating the Update Room function, its status is automatically set to available, allowing a room to be closed for renovations. By using normalisation on the data needed for a Room, a Rates file was created. A rate type is assigned to rooms, a rate amount is assigned to this type therefore if a rate is changed, all the rooms with this type are also changed. This lowers the amount of storage needed for the database and increases its efficiency. By using the Reservations function the members of staff in the hotel can create and save reservations to the Reservations File. While creating a Reservations a Customer is also added to the Customers File which is associated to the Reservation. Therefore, when a payment is processed the Customer’s payment details are retrieved from the Customers file and records of payments are saved to the Payments File. When a Reservation is cancelled the status of the Reservation is changed to cancelled, maintaining relational integrity by not deleting any records from this file. The user can also use the system to check in and check out customers. Upon check in a customer is charged per the business rule. The system also processes no shows, allowing the user to select the reservation that did not show up and charge the customer. How each function interacts with the files is shown in the System Model and the details of each file and how they are linked are demonstrated in the Data Model. Finally, the manager can view reports of the performance of the hotel in a selected year. The user can choose to view these reports over booking per month per type of room and sales amount calculated per month for the selected year. A prototype was created to show the functionality of the system and how the system will validate the information inputted by a user.

# Appendices

## Appendix A – Title

## Appendix B – Title

Might include:

* **Lookup / Reference tables**
* **Sample reports / Listings**