Hotel Reservation Software

Group 1 Project Analysis, Rev 7

CMSC 495 ( 6380) Current Trends and Projects in Computer Science (2205)

University of Maryland Global Campus

Summer 2020

Professor Hung Dao

June 9, 2020

Group 1 Members:

Jacob Valentine

Leonardo Elias

Emmanuel Girin

Marques Young

Revision Table

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision** | **Date** | **Description** | **Contributor** |
| 0 | 06/06/2020 | Creating Document, Cover Page, Revision Table | Jacob Valentine |
| 1 | 06/08/2020 | Editing Format, Added Outside System Section, Defined subsystems, | Emmanuel Girin |
| 2 | 06/09/2020 | Added output data | Leonardo Elias |
| 3 | 06/09/2020 | Finish Subsystem, Subsystem Diagram, Data Processing Steps, Context Diagram | Emmanuel Girin |
| 4 | 06/09/2020 | Added possible risk and risk mitigation | Leonardo Elias |
| 5 | 06/09/2020 | Updated and formatted the input data and output data tables. | Jacob Valentine |
| 6 | 06/09/2020 | Reviewed possible risk and mitigation  Added to possible enhancements | Leonardo Elias |
| 7 | 06/09/2020 | Added to risk and mitigation/ Possible enhancements | Marques Young |

Outside System

The Hotel Reservation Software will be a java application that runs on a client machine. The java application will have a GUI, which a user, usually a hotel staff member, uses to interact with the system. The java application will access a database that is located on the client machine to retrieve information. In future iterations of this software, there is a plan to incorporate a server hosted database. There will be the optional added user, presumably an IT tech, manager, or data analyst, which has special direct access to the database to change records bypassing the java application completely. In this analysis we will focus specifically on one of these outside systems that of the hotel employee, user, which primarily accesses the hotel reservation system to access reservations.

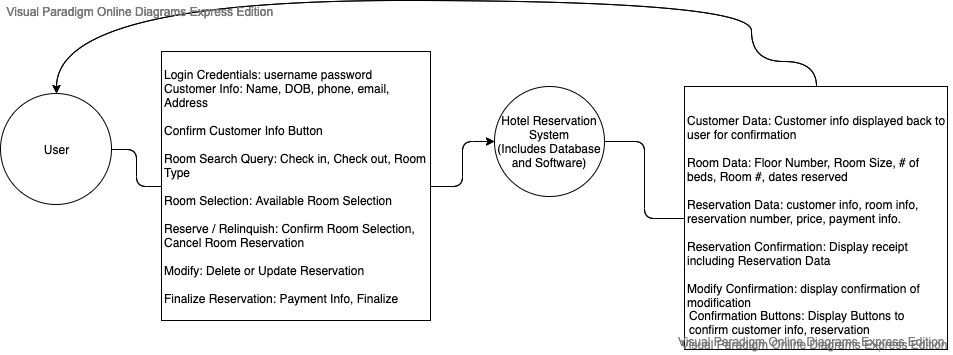
# Input Data

|  |  |  |
| --- | --- | --- |
| Data | Description | Source |
| Login credentials | A username and password associated with a system account. | User |
| Login Verification | A check from the central server saying whether given login credentials match an account on the central server, and what user level and account it corresponds to. | hosting SQL Database |
| Customer info | Personal information used to assign consumers to room reservations, including: first and last name, date of birth, phone #, address, and email. For different functions, only some or all of this information shall be required. | User |
| Room information | Information related to an individual room’s size, location, price, availability, and style, which is used to create new room entries and to modify existing ones. | User, hosting SQL Database |
| Room Search Query | Input information used to search rooms by location, availability, size, and price. | User |
| Room Selection | A selection of a specific room, which a reservation may be placed, updated, or deleted. | User |
| Reserve/Relinquish | The User puts in a request for the system to reserve/relinquish the room, using the provided information. | User |
| Reserve/Relinquish Verification | A check from the server to verify whether the reservation/relinquishment was successful. If it was not, information for why should be provided. | hosting SQL Database |
| Modify Reservation | A request to modify a room reservation. | User |
| Modify Verification | A check from the server to verify whether the modification was successful. If it was not, information for why should be provided. | hosting SQL Database |
| Modify Entry | A request to modify a room entry. | User |
| Delete | A request to delete a room entry. | User |

Output Data

|  |  |  |
| --- | --- | --- |
| Data | Description | Destination |
| Customer Data | A display showing an existing customer’s first and last name, date of birth, phone #, address, and email. | User, hosting SQL Database |
| Room Data | Displayed look up information for an individual room or rooms, including floor number, Room size, number of beds, Room number, dates reserved, and price. | User |
| Reservation Data | A display of the reservation status of given rooms. If the room is reserved, it should display both Customer information linked to the room (first and last name, date of birth, phone #, address, and email), as well as the information of the room itself (Floor number, room number, number of beds, room number and dates reserved).  In addition, the system should display Auto generated information such as a Reservation number, Price of the total reservation, and payment info. | User |
| Reservation Confirmation | Display confirmed reservations.  Email/Print reservation confirmation to customer. | User |
| Modify Confirmation | A confirmation that the reservation has been modified successfully, complete with the updated reservation data. | User |
| Login credentials | A username and password associated with a system account. | hosting SQL Database |

Context Diagram



# Data Processing Steps

At the main screen, a form is displayed where a user must type in a username and password. After entering the information, the user clicks a button to validate the data. The username and password are sent to the UserRecords database for validation. If a failure returns a try again message is displayed. Else progresses to the Customer screen.

The customer screen displays a choice of Creating a New Customer or Existing Customer. The user clicks on either button. If creating a new customer is selected the CreateCustomer screen is displayed. Else the customer Lookup screen is displayed.

* CreateCustomer screen displays an editable form for customer information (see above for the details). Once the user has filled in the information, user clicks a button, then the information is displayed in a locked form, user is asked to continue or edit. If user chooses edit form is redisplayed as unlocked for details to be changed. Else reservation screen displays.
* Customer lookup screen renders a form requesting name, phone number, or email of customer. Once user has entered information in the form, the information is sent to the customerRecords database for confirmation of existing user. If the customer does not exist a negative message will be displayed, and user will either try again or click a button to createNewCustomer. Else the customer information will return display, and the reservation screen displays.

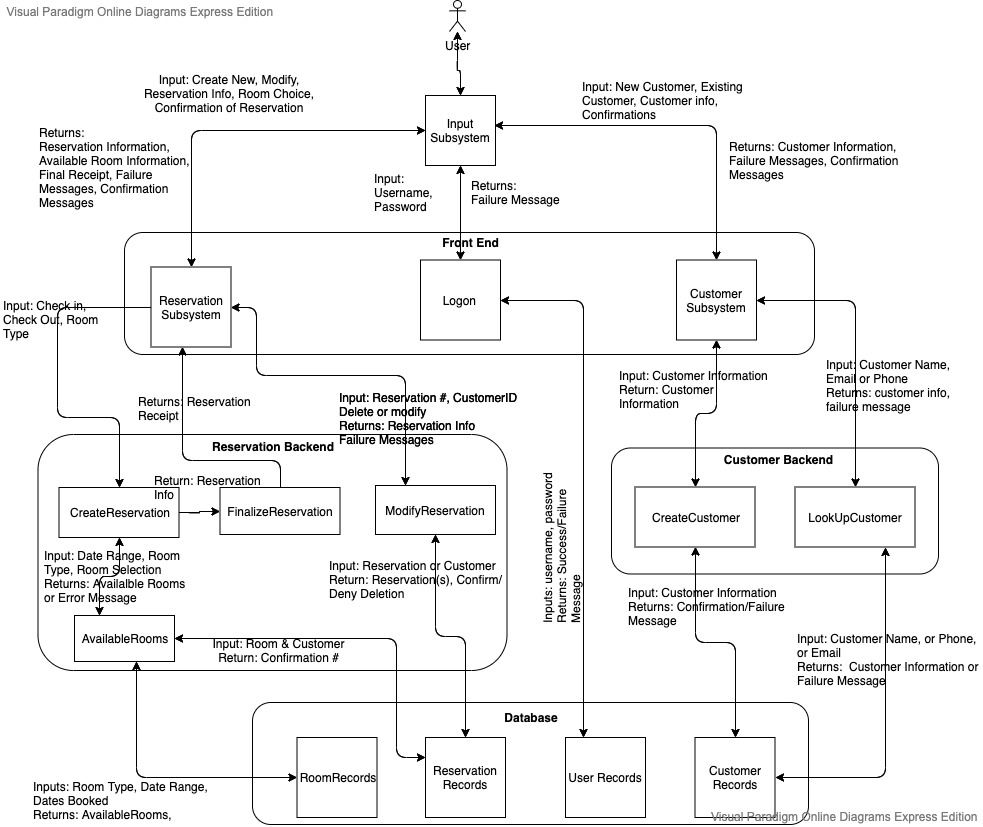
The reservation screen displays the choice of creating new reservation or modifying existing one. If user chooses creating new reservation the Create Reservation screen displays. Else the Modify Reservation screen displays.

* Create Reservation Screen displays a request for Check in Date, Check out Date, Preferred Room Type, and number of Rooms. Once User types in information and clicks confirm, the room data and dates are sent to the Available Rooms subsystem.  
  Available Rooms sends Date Data and preferred room type to RoomRecords. RoomRecords returns the rooms that are available. AvailableRooms displays available rooms to user and price. User selects the room needed. User is prompted to finalize reservation, which sends user to finalize reservation screen. Room information is sent to reservationRecords and roomRecords to update the booking of the room and the reservation data.
  + Modify Reservation Screen displays a form to request reservation confirmation # or perform a lookup of all reservations under customer
  + If user does not have reservation confirmation #, the customer ID info is sent to reservationRecords to return the data of the reservation(s) for that customer.
    - If there are no reservations, a failure message is displayed. User is returned to Create a Reservation screen.
    - Else a list of 1 or more reservations is displayed. User must select from the choice of reservations displayed.
  + If user does have a confirmation # data is sent directly to reservationRecords returning the data of the reservations for that customer. If there are none a failure message is displayed, and user can try again or return to create a Reservation screen.

If a reservation has been found from either of these options above, the user is displayed a choice to delete or modify the reservation. In either case the reservation is deleted from reservationRecords and roomRecords has the availability dates of the room updated. In the case of modifyReservation, the user is sent back to the Create Reservation Screen to make a new Reservation. In the case of delete Reservations, a confirmation is displayed to user that the reservation has been deleted, and the user is sent back to the main reservation screen.

The Finalize Reservation screen appears after a new reservation is created which displays room info, date info, customer info, and price to user in a form, which can be printed for a receipt. Afterwards user is sent back to main screen.

Subsystem Diagram



# Subsystem Descriptions

The following subsystems pass along input and output data. Please refer to the input and output data sections for more specific detailing of the data.

1. Input – Receives input from user involving customer information, reservation information, credentials for logon, and sends output back to the user.
2. Logon – Displays username and login. Takes the input of username and password from user. Displays try again if password is wrong else accesses the Customer subsystem.
3. Customer – Displays existing or new customer. Receives user input. Links to CreateCustomer or LookupCustomer based on choice.
4. CreateCustomer - Displays a request for customer information through a form. Receives user input. Displays data to user for confirmation. If user input does not confirm gives chance to edit customer information form. Upon receiving confirm from user input sends data to Customer Records.
5. LookUpCustomer - Displays request for customer information from user, receives user input, and sends data to CustomerRecords for retrieval of customer data. Displays customer information in form and asks for confirmation of customer. Upon confirmation links to Reservation subsystem.
6. Reservation - Displays new reservation or modify existing reservation. Receives input from user. Links to CreateReservation or ModifyReservation subsystems based on choice
7. CreateReservation - Displays Request for Check In, Check Out, Preferred Room Type. Accepts user input for those values. Sends Data to Available Rooms
8. ModifyReservation - Displays Request for Reservation Info, Receives User Input for either first,last, & Dob, phone number, email, or reservation #. Sends data to ReservationRecords. Displays reservation data. Receive user input for whether to delete or change reservation. If delete send data to ReservationRecords and RoomRecords for update and link to Reservation. Else delete reservation by sending data to ReservationReocrds and RoomRecords, and link to CreateReservation.
9. AvailableRooms - Sends Date Data and Preferred Room Type to RoomRecords. Displays Available Rooms and Price. Accepts user input to select a room & confirms choice. Sends room choice to RoomRecords and ReservationRecords. Send Reservation Data to FinalizeReservation subsystem
10. FinalizeReservation - Displays Reservation Info. User can print receipt information and navigate back to main page.
11. CustomerRecords - Accepts new customer request with customer information data. Creates a new entry in the records. Sends Customer information data back to the calling subsystem. Accepts customer info for look up purposes. Sends back existing customer profile or default failure message.
12. ReservationRecords - Accepts CreateReservation requests with reservation data. Creates new reservation entry. Returns reservation data to subsystem. Accepts ReservationLookup requests with reservation data. Returns existing reservation data back to calling subsystem.
13. RoomRecords - Accepts look up data for room type, checkin, checkout for lookup. Returns available rooms within dates and type. Accepts room # and checkin and check out. Flags the room as booked. Returns confirmation that room is booked.
14. UserRecords - Username and Password File table, updated separately by IT staff. Receives request to confirm correct username and password. Returns confirmation, yes or no of correct username and password.

# Subsystem-Requirement Implementation

|  |  |  |
| --- | --- | --- |
| **#** | **Requirement** | **Subsystem** |
| 1 | Software shall store personal consumer information (first and last name, date of birth, phone #, address, email) | Input, CreateCustomer, CustomerRecords |
| 2 | Software shall protect access to consumer information through username and password access | Input, Logon UserRecords |
| 3 | Software shall use SQL database to store consumer information | Create Customer, CustomerRecords |
| 4 | SQL Database shall contain three tiers of users | UserRecords |
| 5 | SQL Database shall have a master user that has super access to pass off to IT Team | UserRecords |
| 6 | SQL Database shall have an mid-level user that can view and edit reservations / customer data | UserRecords |
| 7 | SQL Database shall have a base-level user that can only view reservations (or maybe mid-level can delete reservation like manager access, and employee base level can only view and make reservations) | UserRecords |
| 8 | Software shall allow a hotel staff to make a reservation for a guest | Create Reservation, Database |
| 9 | Software shall store reservations for rooms on specific dates | Create Reservation, ReservationRecords |
| 10 | Software shall display available rooms on dates | AvailableRooms |
| 11 | Software shall have the capability to potentially display reservations across multiple hotel locations | RoomRecords |
| 12 | Software shall have a basic UI with Menu | Input |
| 13 | Menu shall display option for Finding Existing Reservation, Making new Reservation, deleting reservation and updating reservation. | Input, Reservation, CreateReservation, ModifyReservation |
| 14 | Software shall require customer information to be captured when making a new reservation | Customer, Create Customer, LookUpCustomer |
| 15 | Software shall ask staff it the customer already exists, and perform a look up by name, phone number, or email address | LookUpCustomer |
| 16 | Software shall give Staff option to capture new customer Data if customer does not exist in the database (Register a new customer) | Create Customer |
| 17 | Existing reservations shall be performed by a customer look up | ReservationLookup, LookUpCustomer |
| 18 | Software shall give staff the ability to change a reservation to a new date, room, and location (update reservation) | ModifyReservation |
| 19 | Software shall give staff ability to delete reservation | ModifyReservation |
| 20 | To Update or Create new Reservations: Staff shall enter date, which shall return a list of available room and descriptions | AvailableRooms, CreateReservation, ModifyReservations, FinalizeReservation. |
| 21 | Rooms shall be organized by size (single queen bed, two double beds, single king, suite, and master suite) | RoomRecords |
| 22 | Price of rooms shall generate for each type of room with some randomness built in to simulate dynamic pricing. Later a dynamic pricing software can be linked to this program to tie in reactive pricing to the rooms | RoomRecords, CreateReservation |
| 23 | UI shall be simple to use Graphics UI with text boxes to enter in data and buttons to navigate and perform lookup functions | Input |
| 24 | Software shall ask for Credential Logon at beginning of each use to ensure security | Logon |
| 25 | The software shall allow the user to add extra fees in case of last-minute cancellation or any other reasons. | CreateReservation |

# Possible Enhancements

There are several enhancements that can be added to make for a better reservation process. One of the first enhancements to be made is to give the IT department database privileges in the case that an employee is unable to login with their credentials.

While best practice is least privilege for most employees, managers should be given sole permissions to approve delete reservation requests. This allows the manager to potentially save a sale and/or prevents an employee from unintentionally ruining a customer’s reservation. Managers will also be the only individuals allowed to access customers’ sensitive information.

Existing customers’ information should be left editable in the case that their personal identifiable information was entered incorrectly; this information includes name, phone number, email address, home address, and payment methods.

While the program is built for a small hotel/motel, it does not account for the potential for multiple workstations that have access to booking information. Since this is the case, it is vitally important to check concurrency issues within the database so that rooms are not able to be double booked. This would be something as simple as blacking out rooms that are already booked for certain days.

# Possible Risks and Risk Mitigation

One of the possible risks associated with almost any database or web application is a SQL injection. A SQL injection happens when a third party can use SQL commands to interfere with back-end databases in ways that they shouldn’t be allowed. It generally occurs when websites directly incorporate user-inputted data into an SQL query, the query is then run against a database. There are a few ways to mitigate this type of attack. One of the key ways to mitigate this risk is to use appropriate privileges; don’t connect your database to an account with admin-level privileges unless necessary. You should also get rid of any unnecessary database functionality as this increase “surface size” and makes the database more vulnerable to SQL injections.

Another possible risk is what is known as a brute force attack. This is essentially someone trying to use every password or passphrase until they can gain access to material, they shouldn’t have access to. One of the best practices for preventing a brute force attack is to lockout an account after a set number of attempts, preferably no more than three. A secondary way to prevent a brute force attack is to delay the response time between password attempts. This allows administrators more time to detect potential intrusion.

Multi-factor authentication essentially adds layers to security. Instead of just a username and password, an email address can be used to verify that you are who you say you are; MFA compensates for the weakness of a password or passphrase. MFA also helps reach compliance standards that are established. Having MFA allows for more advanced logins like single sign-on; single sign-on works by validating the user through MFA, once the is authenticated they have access to all the covered apps through the SSO software.

Authentication requirements:

* The password's length must be greater than 8 characters and less than 64 characters.
* Allow all the characters including Unicode and whitespace.
* Implement secure password recovery which allows users to gain access to their account in case they forget their password.