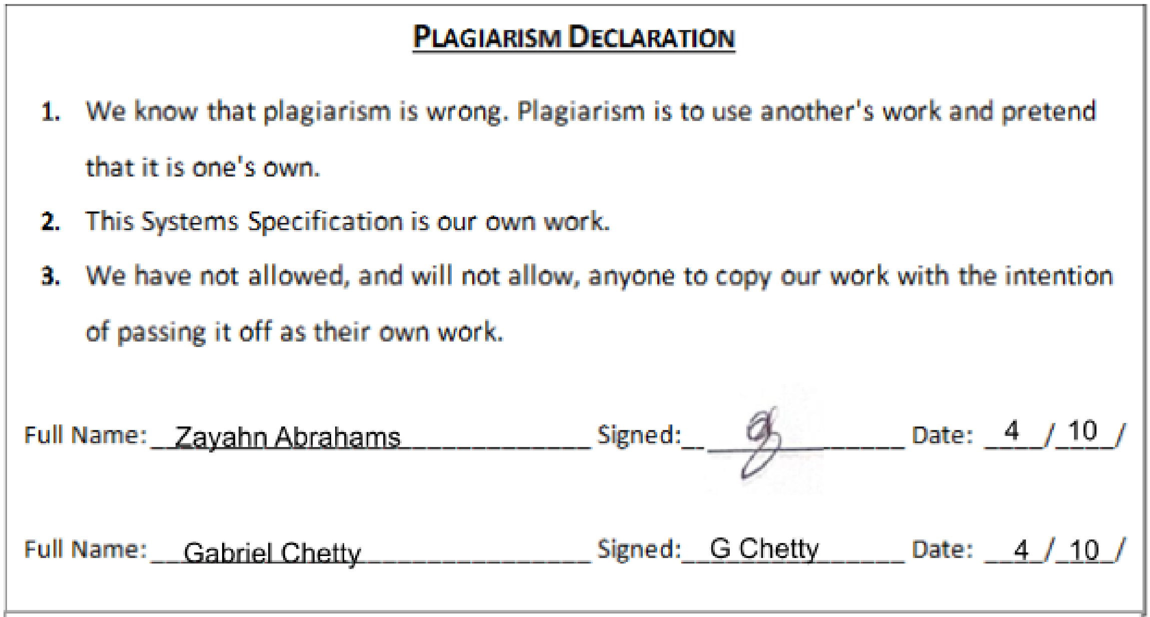
Department of Information Systems

Systems Design & Development

Systems Specification for *Phumla Kamnandi Hotels*

Team Members

Student Number: SRDDAN002	***Student Name: Daniel Sardinha***
Student Number: CHTGAB002	***Student Name: Gabriel Chetty***
Student Number: ABRZAY008	***Student Name: Zayahn Abrahams***
Student Number: RTFMIC002	***Student Name: Micah Retief***



Plagiarism Declaration

1. We know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.
2. This Systems Specification is our own work.
3. We have not allowed, and will not allow, anyone to copy our work with the intention of passing it off as their own work.

Full Name: Zayahn Abrahams Signed: Date: 4 / 10 /

Full Name: Gabriel Chetty Signed: G Chetty Date: 4 / 10 /

Full name: Micah Retief Signed: M Retief Date 4/10/2024

Full name: Daniel Sardinha Signed: D Sardinha Date 4/10/2024

Table Of Content

1. INTRODUCTION

1.1 OVERVIEW OF SPECIFICATION

This document presents a comprehensive design specification for the Phumla Kamnandi Hotels project, aimed at standardizing and improving operational processes within the hotel group. Following the introduction, the document details the following key components: user interface and dialogue design, design sequence diagrams, design class diagrams, entity relationship diagrams, report design, input-output standards and controls, implementation plan, and test plan. The specification references the Phumla Kamnandi test data specification document and case study.

This initiative was launched to resolve inconsistencies across the independently run hotels within the group, as the lack of cohesion between these hotel systems has caused operational inefficiencies and resulted in differing and inconsistent customer experiences.

Six months ago, during the analysis phase, the necessity for an IT solution became evident. This solution aims to integrate and standardise business processes, implement best business practices, optimise service levels, exploit new technologies for tangible business benefits, and deliver management information that is accurate.

The project is in the design and development phase and focuses on establishing a system that will address inefficiencies and meet the necessary requirements. Emphasis has been placed on enhancing customer service and reducing costs. Essential functionalities such as reservation booking, room servicing, and accounts processing have been integrated into a single system, with testing having taken place to refine and improve the system further.

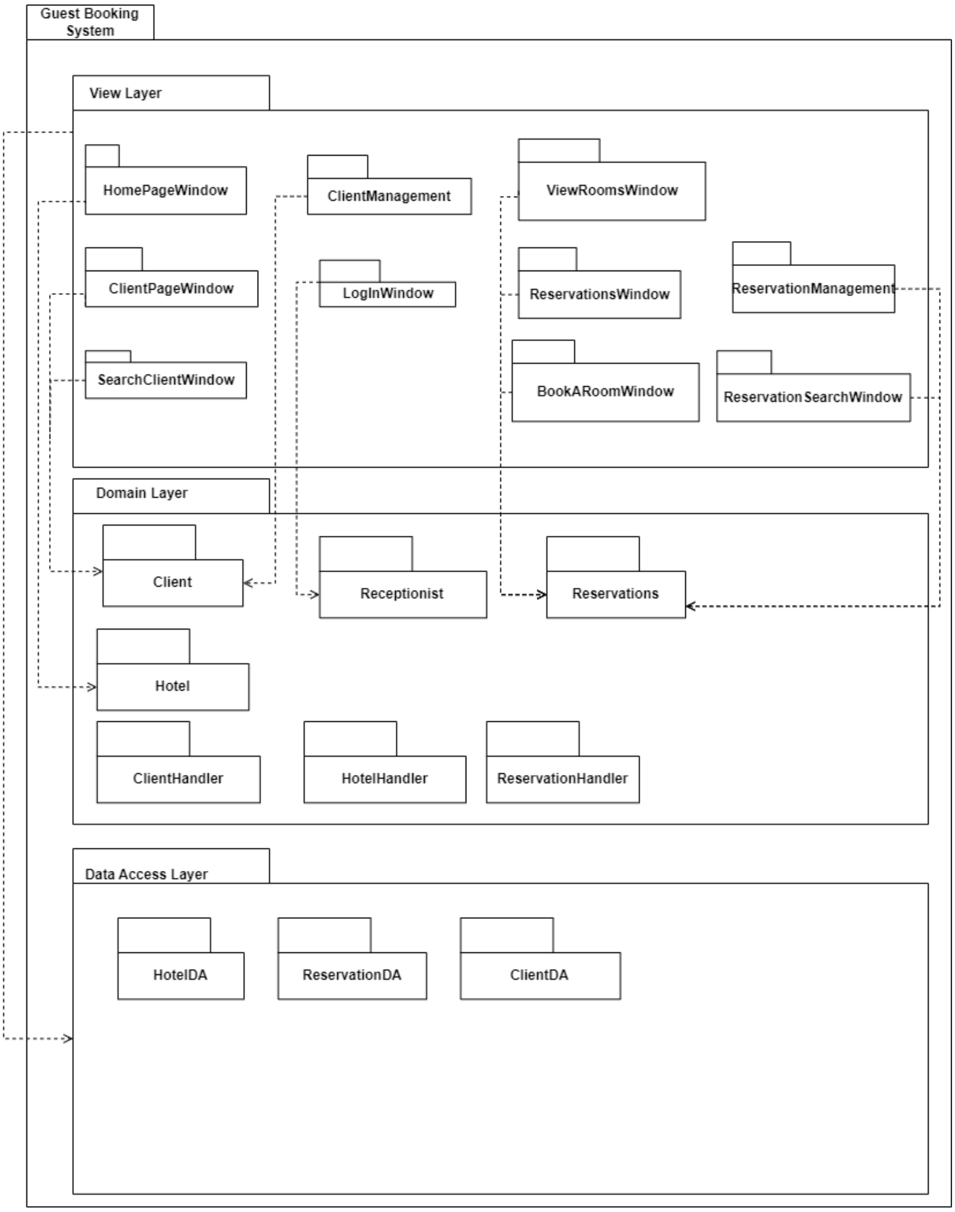
The next steps in the development phase will involve finalising system specifications, improving reporting and control mechanisms, and conducting integration and user tests.

1.2 CONTEXT & SCOPE OF SYSTEM SPECIFICATION

The project seeks to establish a standardised IT system across the Phumla Kamnandi Hotel group to resolve the disconnected nature of its operations. The existing infrastructure varies significantly between the different hotels which has led to inefficiencies and inconsistencies within customer experiences. The new system intends to streamline critical functions such as reservation bookings, guest account management, finance, and HR, ensuring uniform service quality and simplified management of the hotel group.

The scope of this specification is to create and deploy a standardised system that preserves essential flexibility for each hotel. The system must allow a telephone booking to be made, a guest booking to be changed and cancelled and enquiries about guest bookings.

The project's current focus is on the system's core functionalities – namely the management of guest reservations and guest accounts, monitoring of room availability and creation of dynamic reports, which are critical for daily operations. These functionalities ensure that the key functions are operating optimally and help to prevent overbooking. It is essential that critical systems are addressed first before additional functionalities can be implemented in order to ensure usability and functionality.



1.3 Design Assumptions & Constraints

Constraints

- Technical constraints: The hotels in the hotel group utilise a diverse range of systems so it may be difficult to create a standardised solution. The diversity of the IT infrastructure used by the hotels calls for a careful selection of compatible technologies that work on both older and newer systems. The design must ensure interoperability and integration with existing systems.

- Scalability: The system must be designed with scalability in mind, allowing for future growth of the hotel group without requiring significant redesign. This includes the potential for incorporating additional features such as loyalty programs, cable TV, and conferencing functionalities as the needs of the hotel evolve.
- Performance constraints: The system must be able to handle large volumes of data to ensure optimal performance to support high demand, especially during peak seasons.

Assumptions

- Existing data can be transferred from the previous systems to the new system without data loss or corruption, allowing for continuity in business operations and data integrity.
- All hotels offer one standard room and there are no variations for suites or children's rooms, simplifying the design of reservation booking management.
- Staff at the hotels will receive training to familiarize themselves with the new system which will allow for a seamless transition between systems.

Trade-off Analysis

Resource Use vs Productivity: The project team has conducted a trade-off analysis with regards to resource allocation versus productivity outcomes. Employing more developers could accelerate the development process but it could also increase costs. The team must balance resource allocation to maintain productivity and adhere to budget constraints simultaneously.

2. USER INTERFACE & DIALOGUE DESIGN

In this section, we will explore the design and functionality of the user interface and dialogue for our hotel reservation system. We have structured each screen to ensure clarity and ease of use, with separate sections that handle different elements, such as functionalities like client management, reservations, and reporting. Additionally, we will define all data elements associated with each screen, including their edit criteria and data entry controls, to ensure complete data integrity on our side to prevent errors and "bad data" from being inputted to the system. Miscellaneous messages, access restrictions, and security considerations to maintain our systems integrity, are all addressed in this section.

Layout of GUI's:

We have clearly laid out each screen or window in the hotel reservation system with distinct sections for different functionalities. For example, the "Clients" tab allows for adding new clients, while the "Reservations" tab is used for booking rooms and managing reservations. **We've also used "Guests" and "Clients" interchangeably as well as "Bookings" and "Reservations"**

Graphic Representation of Each Interface:

Our various forms show:

Login Screen: Includes fields for username and password with a login button. This only allows for receptionists and admins to login to the system

Client Management Screen: Features fields for adding new clients, including name, email, and phone number. As well as including functionality to search for existing clients

Reservation Management Screen: Contains fields for entering client phone numbers, room IDs, and date selections for booking rooms. As well as Viewing/Editing/Deleting said bookings

Reports Screen: It displays graphs and reports for reservation occupancy and expected monthly revenue.

Data Elements Definition:

Each screen has specific data elements, such as:

Login Screen: Username (alphanumeric, mandatory), Password (alphanumeric, mandatory).

Client Management Screen: Name (alphanumeric, mandatory), Email (alphanumeric, mandatory), Phone (numeric, mandatory).

Reservation Management Screen: Client Phone Number (numeric, mandatory), Room ID (numeric, mandatory), Start Date (date, mandatory), End Date (date, mandatory), Deposit (True/False, optional), Room Service (True/False, optional).

Reports Screen: Date Range (date, mandatory), Revenue (numeric, calculated).

Data Entry Controls:

To prevent edit bypassing, the system should implement:

- Mandatory fields that must be filled before submission.
- Validation checks for data types (e.g., numeric, alphanumeric).
- Range checks for dates to ensure logical booking periods.

This ensures that all data properly collected without causing errors

Miscellaneous Messages:

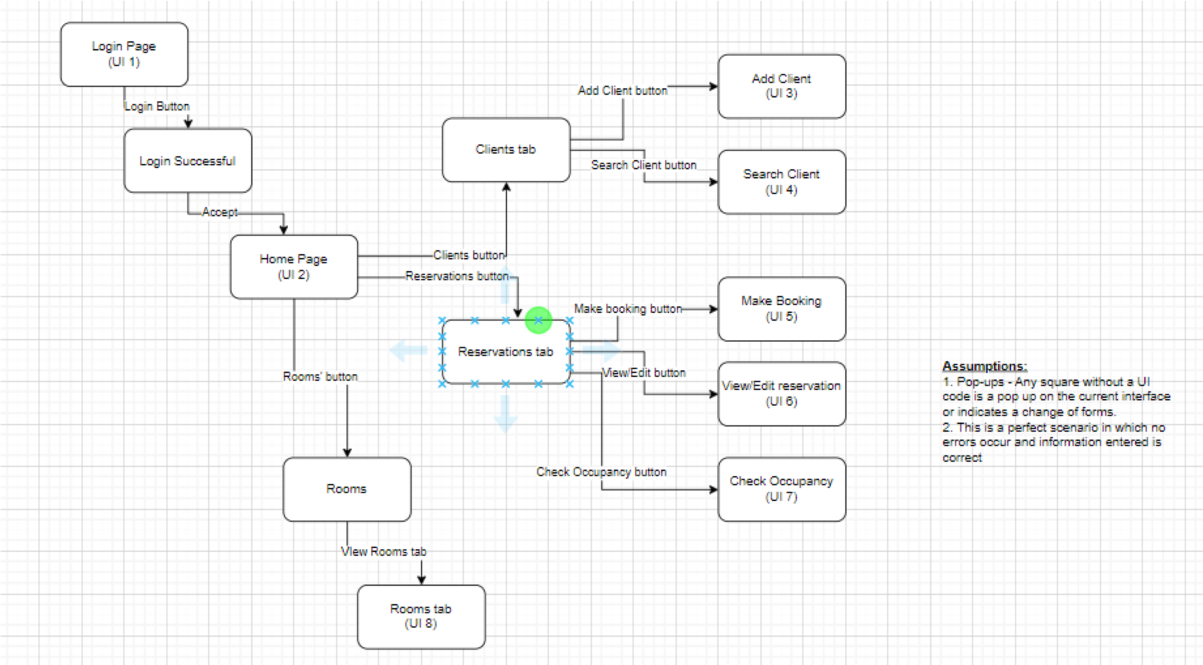
The system should provide feedback messages such as:

- Error Messages: For invalid inputs or missing compulsory fields.
- Confirmation Messages: For successful data entries or updates.

Access Restrictions and Security:

Access to different sections of the system should be restricted based on user roles. For example, only authorized personnel should have access to the “Reports” section in the Reservations tab. In our case only admins and receptionists are able to access the system.

Interface Flow Diagrams



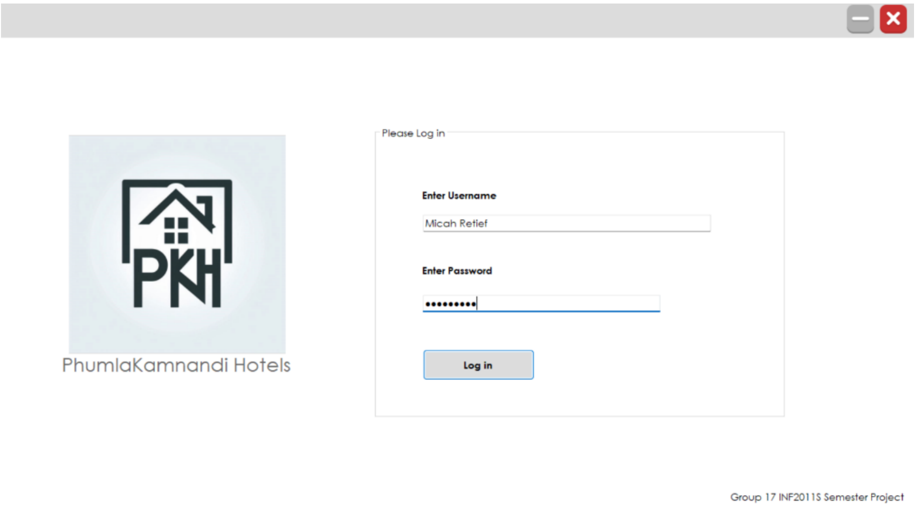
Assumptions:

1. Pop-ups - Any square without a UI code is a pop up on the current interface or indicates a change of forms.
2. This is a perfect scenario in which no errors occur and information entered is correct

Screen Standards

Our UI design follows a clean, consistent layout with a clear navigation bar at the top, ensuring easy access to key sections like "Home”, "Clients”, and "Reservations." The grid like system organizes content with ample whitespace, making the interface uncluttered and easy to navigate for new users. A soft pastel colour scheme with high-contrast text ensures readability, while accent colours highlight our important interactive elements like buttons and tabs. Each screen, from client management tab to room booking, follows a logical and intuitive flow with simple, well-labelled forms and responsive design, creating a modern, professional, and user-friendly experience that we quite like.

Detailed Screen Layout

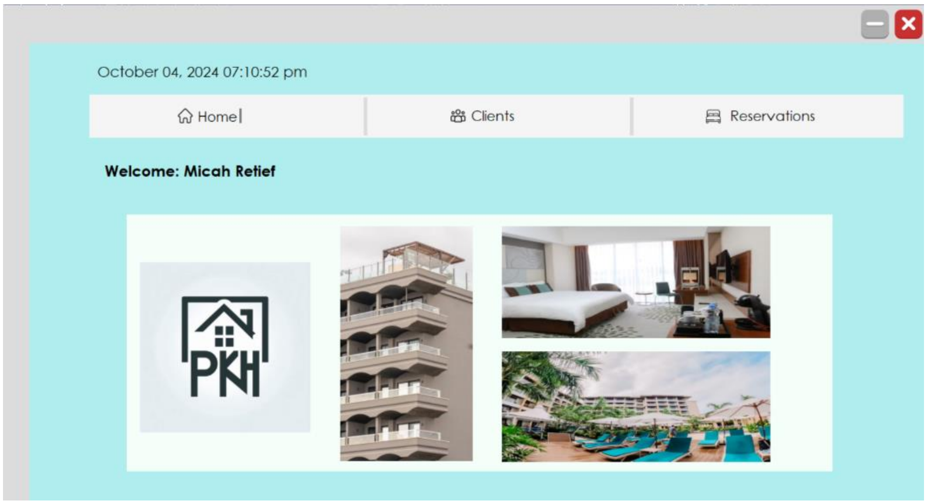


Enter Username

Micah Retief

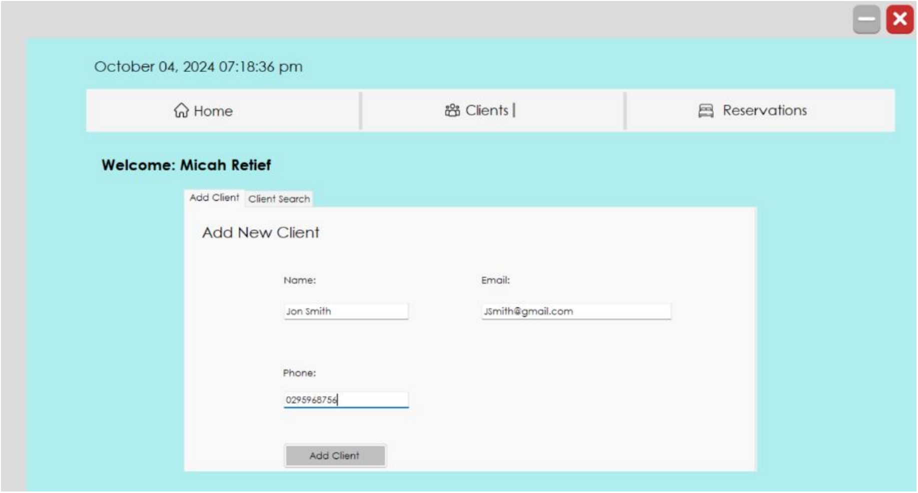
Enter Password

•••••••••



October 04, 2024 07:10:52 pm

Welcome: Micah Retief



October 04, 2024 07:18:36 pm

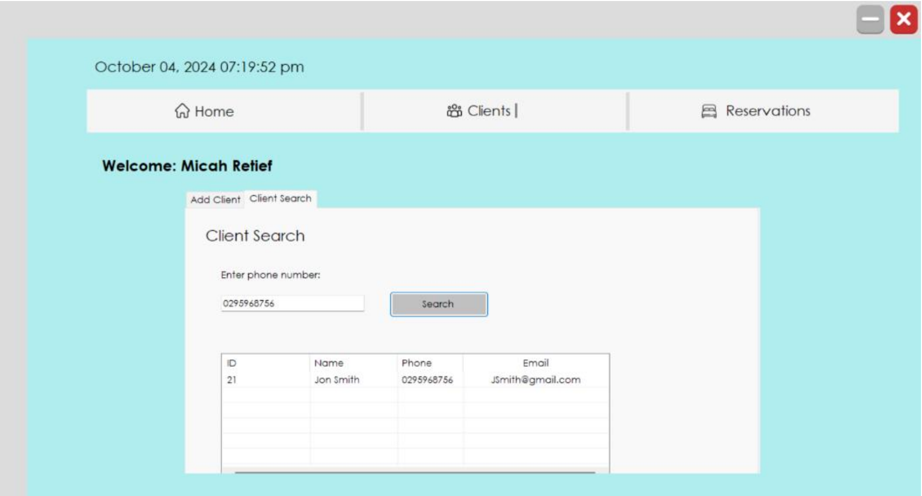
Welcome: Micah Retief

Add New Client

Name: Jon Smith

Email: JSmith@gmail.com

Phone: 0295968756



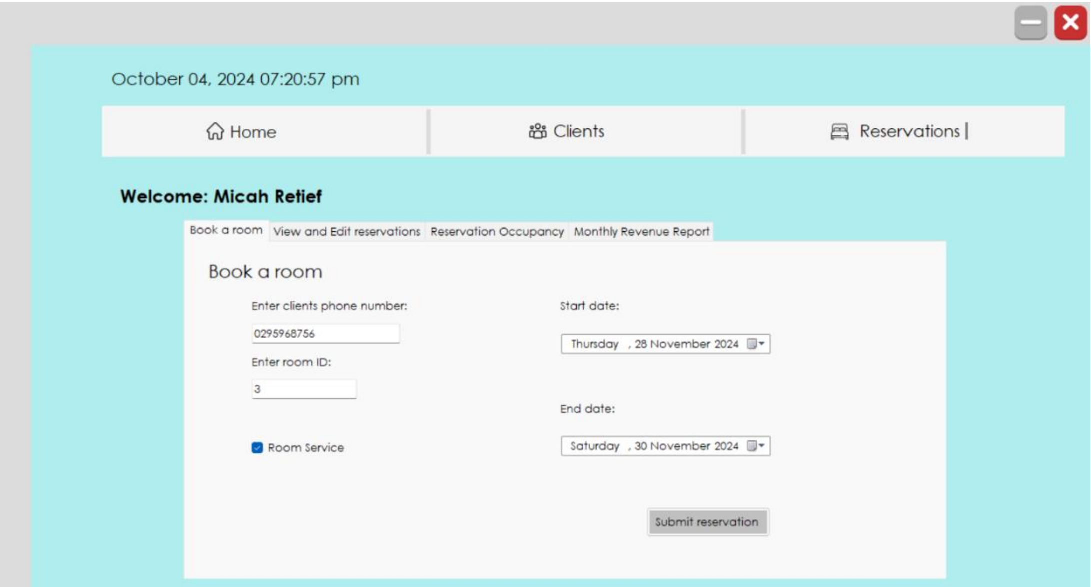
October 04, 2024 07:19:52 pm

Welcome: Micah Retief

Client Search

Enter phone number: 0295968756

ID	Name	Phone	Email
21	Jon Smith	0295968756	JSmith@gmail.com



October 04, 2024 07:20:57 pm

Welcome: Micah Retief

Book a room

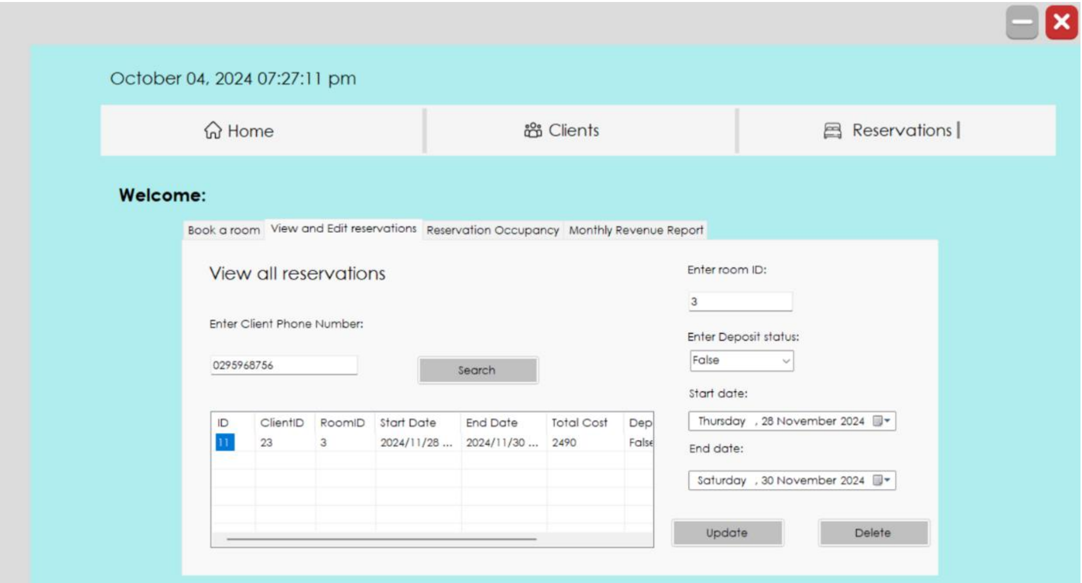
Enter clients phone number: 0295968756

Enter room ID: 3

☑ Room Service

Start date: Thursday , 28 November 2024

End date: Saturday , 30 November 2024



October 04, 2024 07:27:11 pm

Welcome:

View all reservations

Enter Client Phone Number: 0295968756

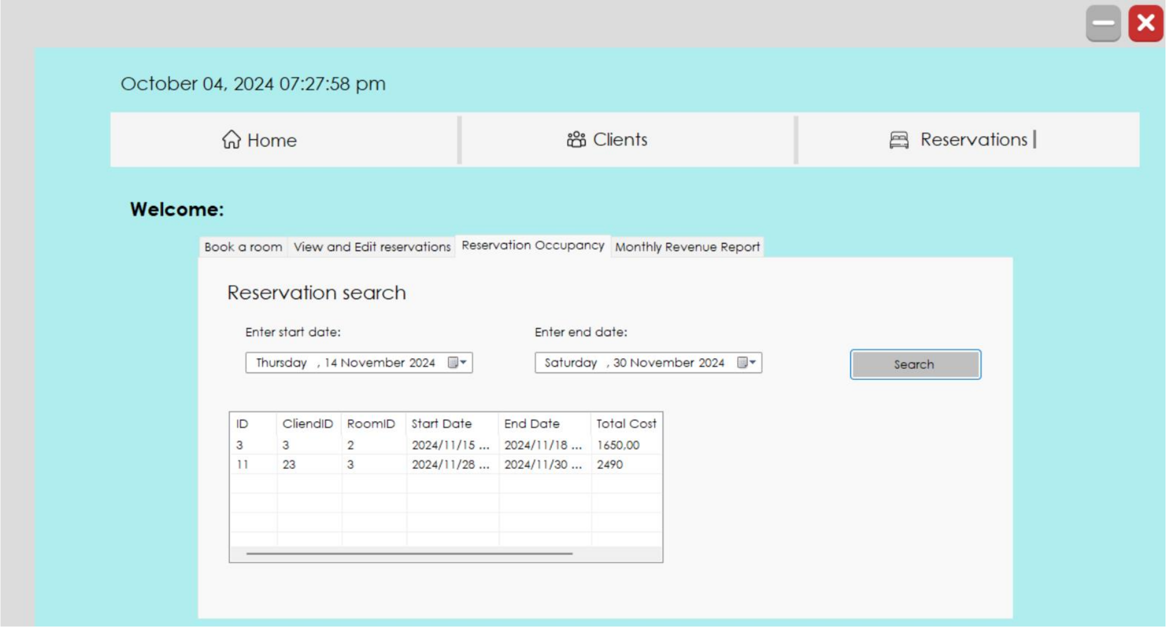
ID	ClientID	RoomID	Start Date	End Date	Total Cost	Dep
11	23	3	2024/11/28 ...	2024/11/30 ...	2490	False

Enter room ID: 3

Enter Deposit status: False

Start date: Thursday , 28 November 2024

End date: Saturday , 30 November 2024



October 04, 2024 07:27:58 pm

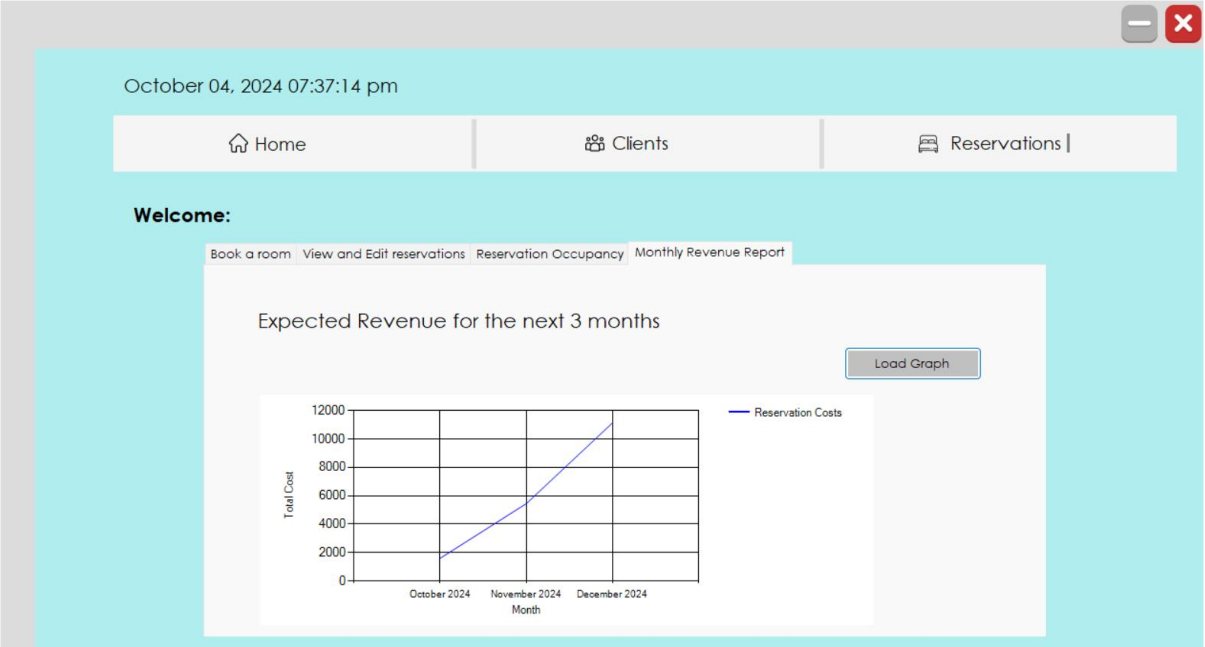
Welcome:

Reservation search

Enter start date: Thursday , 14 November 2024

Enter end date: Saturday , 30 November 2024

ID	CliendID	RoomID	Start Date	End Date	Total Cost
3	3	2	2024/11/15 ...	2024/11/18 ...	1650.00
11	23	3	2024/11/28 ...	2024/11/30 ...	2490



October 04, 2024 07:37:14 pm

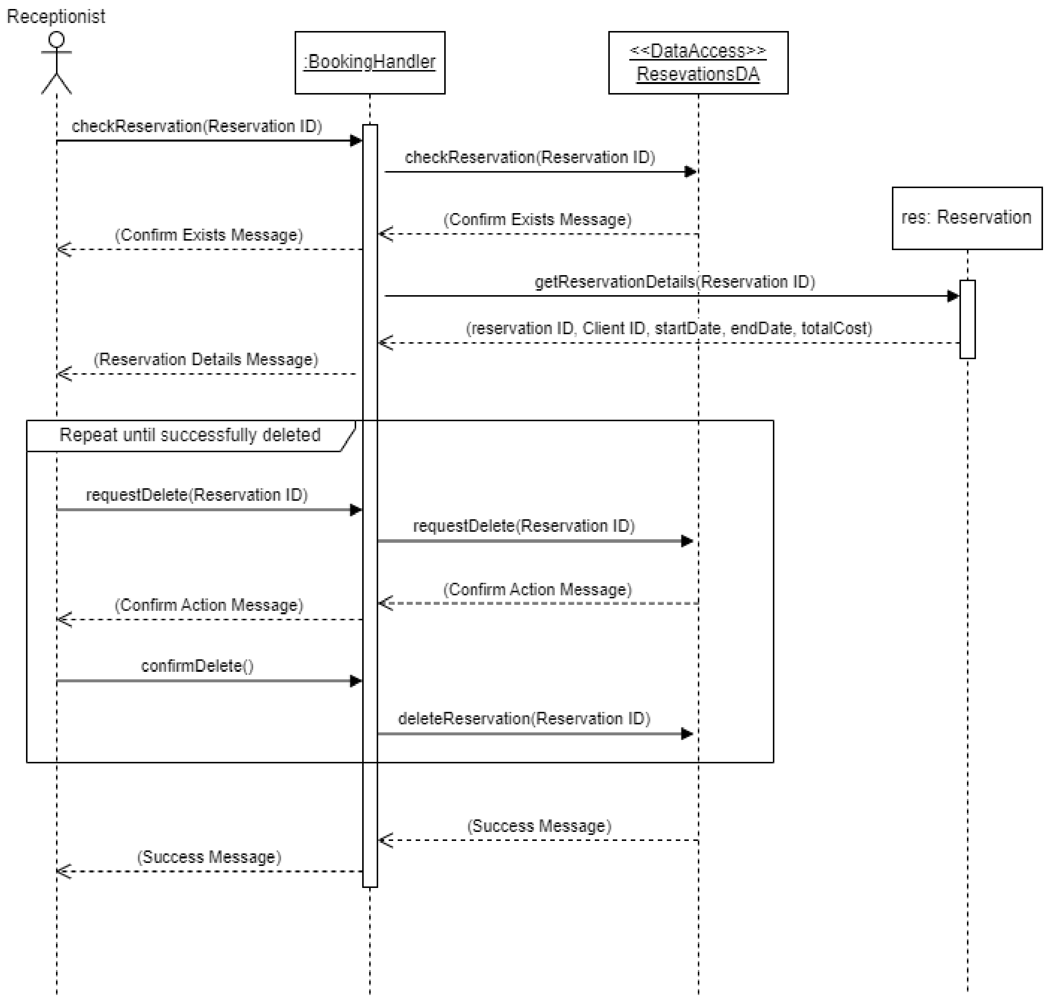
Welcome:

Expected Revenue for the next 3 months

3. DESIGN SEQUENCE DIAGRAMS

3.1. CANCEL A GUEST BOOKING

This diagram models the receptionist's interaction with the system when cancelling a booking. The system first confirms that the booking to be cancelled exists. When it is confirmed, the receptionist will request the booking to be deleted. Once the booking is successfully deleted, a message will be returned to the receptionist to confirm the successful deletion.


Receptionist
:BookingHandler
<<DataAccess>>
ResevationsDA
res: Reservation
checkReservation(Reservation ID)
checkReservation(Reservation ID)
(Confirm Exists Message)
(Confirm Exists Message)
getReservationDetails(Reservation ID)
(reservation ID, Client ID, startDate, endDate, totalCost)
(Reservation Details Message)
Repeat until successfully deleted
requestDelete(Reservation ID)
requestDelete(Reservation ID)
(Confirm Action Message)
(Confirm Action Message)
confirmDelete()
deleteReservation(Reservation ID)
(Success Message)
(Success Message)

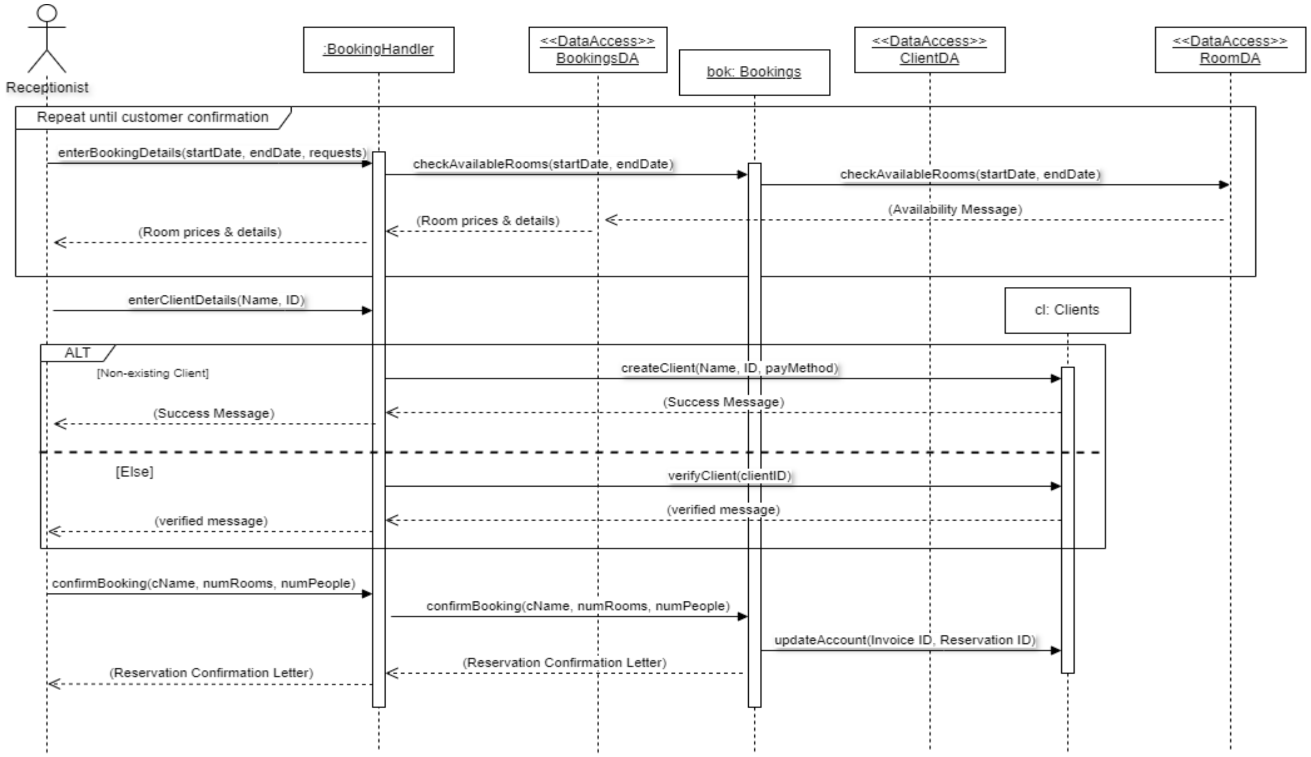
Assumptions:

Audit Log: Assume the system stores deleted reservations for a period of time.

Existence assumption: Assume that there are existing reservations to be deleted.

3.2. MAKE A TELEPHONE BOOKING

This diagram models the receptionist receiving a telephone booking, they record the details and inform the customer of the costs. They repeat this until the customer is happy with the booking and its associated costs. The client's account is then updated to show the booking, and lastly the booking is completed. The booking confirmation letter is returned to the receptionist once the booking is completed.


Receptionist
:BookingHandler
<<DataAccess>>
BookingsDA
bok: Bookings
<<DataAccess>>
ClientDA
<<DataAccess>>
RoomDA
Repeat until customer confirmation
enterBookingDetails(startDate, endDate, requests)
checkAvailableRooms(startDate, endDate)
checkAvailableRooms(startDate, endDate)
(Availability Message)
(Room prices & details)
(Room prices & details)
enterClientDetails(Name, ID)
cl: Clients
ALT
[Non-existing Client]
createClient(Name, ID, payMethod)
(Success Message)
(Success Message)
[Else]
verifyClient(clientID)
(verified message)
(verified message)
confirmBooking(cName, numRooms, numPeople)
confirmBooking(cName, numRooms, numPeople)
updateAccount(Invoice ID, Reservation ID)
(Reservation Confirmation Letter)
(Reservation Confirmation Letter)

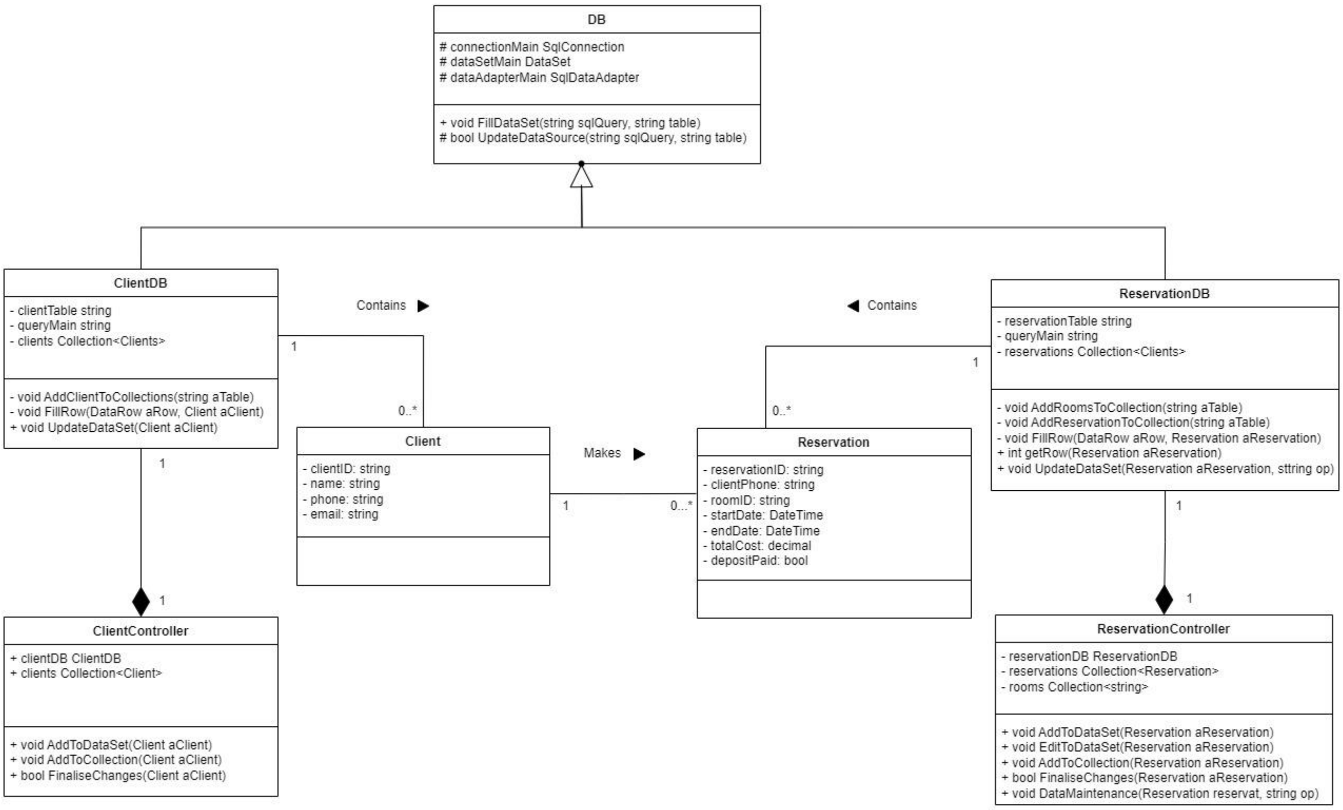
Assumptions:

Perfect Solution: Assumption: Assume no exeptional conditions.

Availability assumption: Assume that there are available rooms.

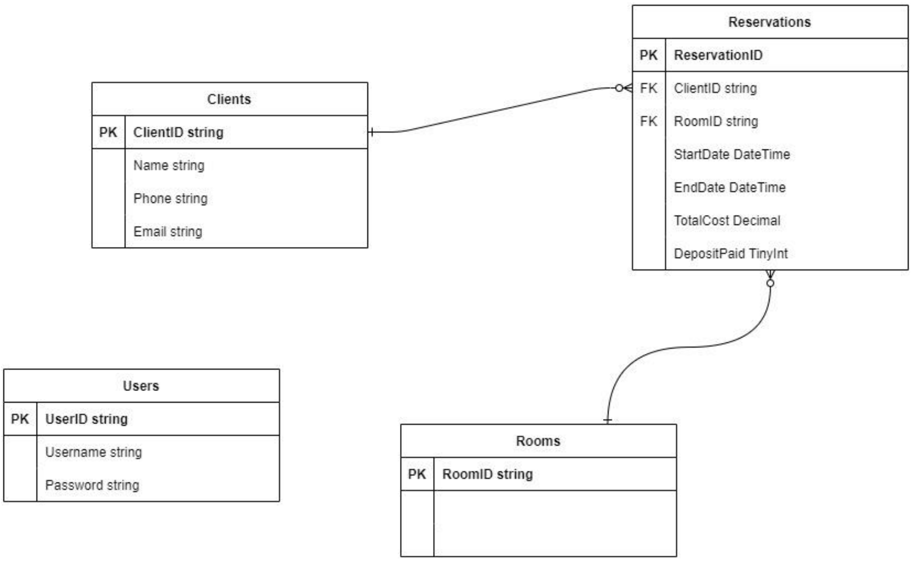
4. Design Class Diagrams

The design class diagram shows the relationships between the classes within the hotel booking system. The DB class is the only class that interacts directly with the database and has two subclasses, ClientDB and ReservationDB which use the DataAdaptor, Connection and DataSet of the DB class. ClientDB and ReservationDB holds the collection of Client and Reservation objects respectively and contain methods which offer CRUD functionality to our program (Uses methods in the DB class). The ClientController and ReservationController classes are crucial in connecting our UI to our database and ClientDB and ReservationDB both solely exist and are utilised within their respective controller classes. The controller classes hold a reference to their respective DB collections as well and contain additional methods to help finalise changes made by the user.


DB
# connectionMain SqlConnection
# dataSetMain DataSet
# dataAdapterMain SqlDataAdapter
+ void FillDataSet(string sqlQuery, string table)
# bool UpdateDataSource(string sqlQuery, string table)
ClientDB
- clientTable string
- queryMain string
- clients Collection<Clients>
- void AddClientToCollections(string aTable)
- void FillRow(DataRow aRow, Client aClient)
+ void UpdateDataSet(Client aClient)
Contains ▶
1
0..*
Client
- clientID: string
- name: string
- phone: string
- email: string
Makes ▶
1
0...*
Reservation
- reservationID: string
- clientPhone: string
- roomID: string
- startDate: DateTime
- endDate: DateTime
- totalCost: decimal
- depositPaid: bool
◀ Contains
0..*
1
ReservationDB
- reservationTable string
- queryMain string
- reservations Collection<Clients>
- void AddRoomsToCollection(string aTable)
- void AddReservationToCollection(string aTable)
- void FillRow(DataRow aRow, Reservation aReservation)
+ int getRow(Reservation aReservation)
+ void UpdateDataSet(Reservation aReservation, sttring op)
1
1
ReservationController
- reservationDB ReservationDB
- reservations Collection<Reservation>
- rooms Collection<string>
+ void AddToDataSet(Reservation aReservation)
+ void EditToDataSet(Reservation aReservation)
+ void AddToCollection(Reservation aReservation)
+ bool FinaliseChanges(Reservation aReservation)
+ void DataMaintenance(Reservation reservat, string op)
1
1
ClientController
+ clientDB ClientDB
+ clients Collection<Client>
+ void AddToDataSet(Client aClient)
+ void AddToCollection(Client aClient)
+ bool FinaliseChanges(Client aClient)

5. Entity Relationship Diagram

The database of our system contains 4 tables. One houses the login credentials of our receptionists, another holds all our data for our clients, one holds all available room IDs, and the last one has all our reservation details. This DBMS model is illustrated within the Entity Relationship model below and details of each attribute is given in the below data dictionary table.

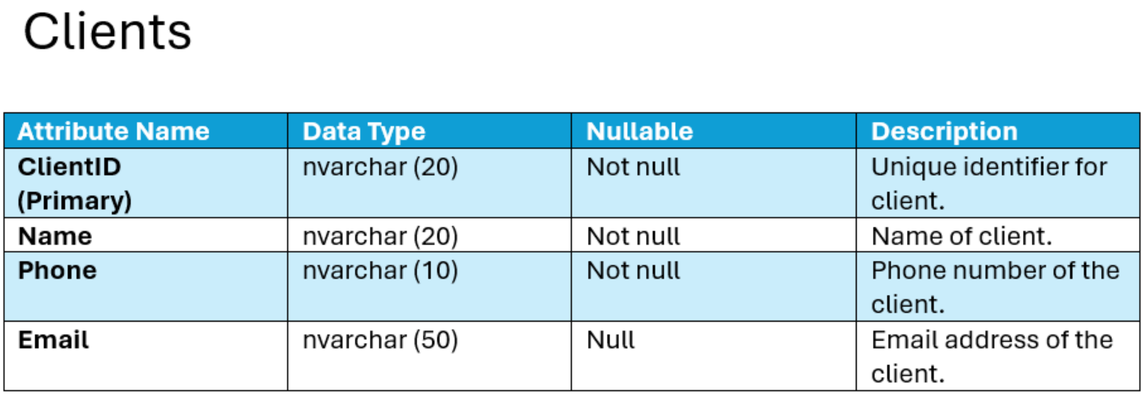


Clients	
PK	ClientID string
	Name string
	Phone string
	Email string

Reservations	
PK	ReservationID
FK	ClientID string
FK	RoomID string
	StartDate DateTime
	EndDate DateTime
	TotalCost Decimal
	DepositPaid TinyInt

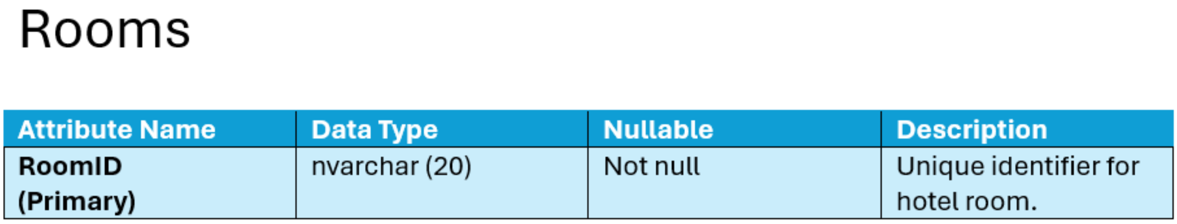
Users	
PK	UserID string
	Username string
	Password string

Rooms	
PK	RoomID string



Clients

Attribute Name	Data Type	Nullable	Description
ClientID (Primary)	nvarchar (20)	Not null	Unique identifier for client.
Name	nvarchar (20)	Not null	Name of client.
Phone	nvarchar (10)	Not null	Phone number of the client.
Email	nvarchar (50)	Null	Email address of the client.



Rooms

Attribute Name	Data Type	Nullable	Description
RoomID (Primary)	nvarchar (20)	Not null	Unique identifier for hotel room.

Reservations

Attribute Name	Data Type	Nullable	Description
ReservationID (Primary)	nvarchar(50)	Not null	Unique identifier for the reservation.
ClientID (Foreign)	nvarchar (20)	Not null	Foreign key for client identifier.
RoomID (Foreign)	nvarchar (20)	Not null	Foreign key for room identifier.
StartDate	DateTime	Not null	Start date of reservation.
EndDate	DateTime	Not null	End date of reservation.
TotalCost	decimal(12,2)	Not null	Total cost of the reservation.
DepositPaid	TinyInt	Not null	Indicator on whether deposit has been paid or not. 1= yes, 0 = no.
RoomService	TinyInt	Not null	Indicator on whether room service is selected or not. 1= yes, 0 = no.

Users

Attribute Name	Data Type	Nullable	Description
UserID (Primary)	nvarchar (5)	Not null	Unique identifier of the receptionist.
Username	nvarchar(40)	Not null	Username of the receptionist.
Password	Nvarchar(50)	Not null	Password of the receptionist.

6. Report Design

This section outlines the key reports that the hotel system will generate, providing critical insights into operations and financial projections, of which such information can heavily alter the path the business takes in the foreseeable future.

The system will offer two essential reports: one focused on occupancy levels, which will be dynamically generated based on specified dates selected by the receptionist, and another that forecasts the expected monthly revenue for the next three months. Both reports are designed to support informed decision-making by offering valuable, actionable data. Detailed requirements and functionalities for each of our reports are specified in the following subsections provided.

6.1 Report 1 – Occupancy Report

6.1.1. ***Detailed Output Requirements***

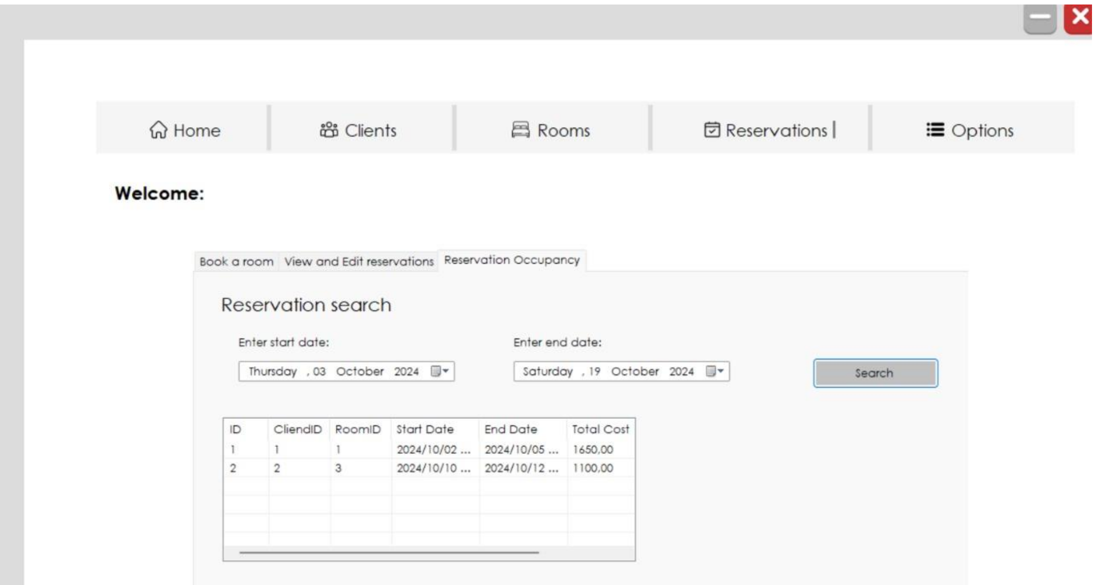
The Occupancy Level Report aims to provide an overview of the hotel's room availability for specific periods of time. The report plays a key role in managing room availability and ensuring that the hotel operates at an ideal capacity while avoiding overbookings. Receptionist staff use this report to monitor current and upcoming room availability, making it especially useful during busy periods such as holidays, peak travel seasons or large events.

Output type & ID:	**Electronic Report (ID: OR-001)**
Report objectives:	The report aims to provide an overview of the hotel's room availability to assist management in planning for peak periods, identify room availability and ensure optimal hotel room occupancy.
Audience:	The audience are the managers, receptionist staff, housekeeping staff, Exco and CRM team.

Content:	The report's content includes a date range that the report has been generated for, the ID number, the Client ID, the Room ID, the booking results and their start and end date and the total cost.
Layout:	The heading "Reservation search" is at the top of the screen, two boxes to enter the start and end date of the reservation search are located below, and there is a search button as well as a table showing the ID, Client ID, Room ID, start and end date of the booking and its total cost.
Selection:	The date ranges for which the room availability data is required can be selected.
Sequence:	The reservation data is displayed by date in ascending order, starting with the earliest date.
Comparison:	The report should have summarised data about occupancy from the same period in a previous year. This provides valuable context into the potential performance of the upcoming season.
Grouping/Summarisation:	The reservation data is grouped by a time period which can be selected by the user.
Media to be used:	A dynamic on-screen display is to be used.

Frequence, timing, delivery:	The report is generated on-demand. It can be generated daily for receptionists to monitor room availability. It is delivered electronically and displayed on the hotel's system.
Distribution:	The report is shared within the hotel's management team as well as its head office via the hotel management system. It is made available to reception, the hotel management team, and housekeeping staff.
Privacy, security, and integrity requirements:	The report is accessible only to authorised staff members. This includes receptionists, managers, and the head office. Access control measures should ensure that guests' information is not exposed to ensure privacy and there should be regular integrity checks to certify that the data has not been altered by unauthorised persons.

6.1.2 ***Report Layout:***



Reservation search

Enter start date: Thursday , 03 October 2024

Enter end date: Saturday , 19 October 2024

ID	CliendID	RoomID	Start Date	End Date	Total Cost
1	1	1	2024/10/02 ...	2024/10/05 ...	1650,00
2	2	3	2024/10/10 ...	2024/10/12 ...	1100,00

6.2 Report 2 – Revenue Report

6.2.1 ***Detailed Output Requirements***

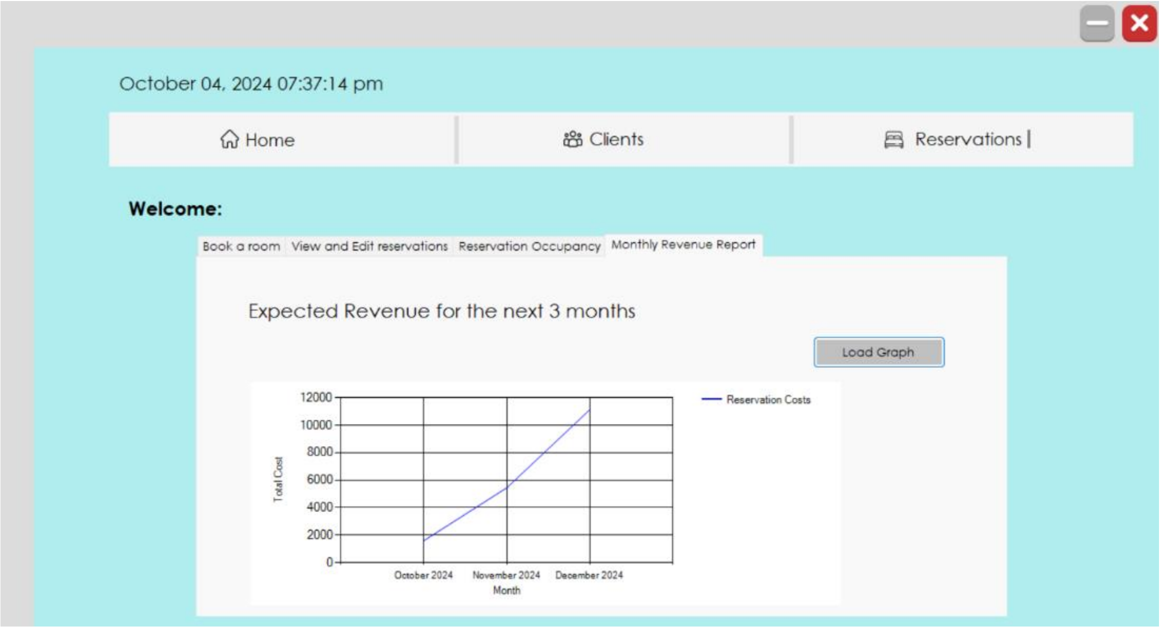
The Monthly Revenue Report aims to track and summarize the total revenue generated by one of the hotels for a specific period. This report assists hotel management in assessing financial performance and identifying trends so that hotel operations can be optimised. The report is essential for monitoring the financial wellbeing of the hotel, determine peak seasons and slow periods and making informed decisions regarding hotel expenditure.

Output type & ID:	**Electronic report, ID: MRR-001**
Report objectives:	The report’s objective is to provide a clear overview of the hotel’s monthly revenue. This enables informed decision making about budgeting and resource allocation.
Audience:	The audience is hotel management, investors, and key stakeholders.

Content:	The report's content includes the total reservation costs graphed against the months of October, November, and December 2024. This provides an overview of the hotel's performance over a specified period so that peak and slow periods can be easily identified.
Layout**:**	A line graph format with Total Reservation Cost on the y-axis and Month on the x-axis. A Load Graph button generates the graph, and the heading "Revenue" is located above the line graph.
Selection:	The receptionist selects a date range.
Sequence:	On the graph, Months and Total Cost are listed in ascending order.
Comparison:	The report should compare the total revenue for the current month with total revenue for some previous month to provide insight into the business performance from the period.
Grouping/Summarisation:	The data is grouped in a period of 3 months. The graph shows the change in total revenue cost (in R1000 intervals on the y-axis) over the course of three months (on the x-axis).
Media to be used:	An electronic display is presented on the hotel management system.
Frequency, Timing, Delivery:	The report is generated on-demand. The report will typically be generated every three months

	to monitor trends over the season.
Distribution:	The report is available to hotel management, the finance department, key investors, and stakeholders.
Privacy, security, and integrity requirements:	User authentication will be required so that only authorised users can access the information. Regular audits should be required to ensure data integrity. It is essential that access logs note who has accessed the report.

6.2.2 ***Report Layout***



October 04, 2024 07:37:14 pm

Expected Revenue for the next 3 months

7. I/O Standards & Controls

The detailed design of the system's input and output is outlined in this section, focusing on the user interactions within the hotel booking system. It provides a comprehensive description of the various UI screens, reports, and validation methods implemented to ensure data integrity, user control, and output accuracy across the board. Our system has been developed with a focus on providing clear, structured outputs such as reservation confirmations, occupancy reports, and cancellation screens, while also incorporating plentiful robust validation mechanisms to minimize errors and improve the overall user experience. Additionally, input integrity controls, automated dynamic calculations, and strict output integrity measures have been implemented to ensure that the system operates reliably and consistently. And it does so, according to our tests. These features ensure that all critical data is accurately displayed and maintained, enhancing operational efficiency and decision-making capabilities.

7.1. Formalised Outputs:

Reservation Confirmation Screen:

Once a valid reservation has been entered, the system will display the following details:

- Reservation ID – The booking's unique ID
- Client Details – Names, contact details and any special requirements.
- Reservation Period – The start and end dates of the booking.
- Payment Total – A breakdown of the total cost of the booking.

Occupancy Report:

This should provide an overview of the occupancy within one of the hotels and allow management to plan for peak seasons and to assist with business decisions.

- Generation Date – The date that the report was requested.
- Reporting Period – The period that the report covers.
- Availability – Show the number of available rooms for the given period.
- Occupancy Graph – Show the occupancy percentage per day.
- Housekeeping status – State the cleaning status of each available room.

Reservation Cancellation Screen:

When a receptionist cancels a reservation, the system will display a cancellation confirmation screen. This would help to prevent misunderstandings and double bookings.

- Reservation ID – The booking's unique ID.
- Client Details – Names, contact details, special requirements.
- Reason for Cancellation – An optional input for the system if there is a particular reason for the cancellation.

7.2. Built-In Validation to Ensure Requirements are Met

In our system we ensured data integrity and limited user errors by using several built-in validation methods. This needed to be done while balancing the business needs with the operational needs.

Client Validation:

The primary fields are the Client name and phone number, the email address is an optional field.

- Phone number validation – Done by ensuring that the number entered is a 10-digit number.
- Name validation – Done by ensuring that the name entered consists of only valid characters, i.e. no numbers are permitted.

User Control Validation:

The user of the system needs to be a valid user stored within the system. It requires that the user enters an existing username and password.

- User Validation – The system checks the Users database to determine whether the entered username and password is valid. Only continuing to the main system once they are valid.

7.3. Input Integrity Controls

Input integrity requires that we have methods for ensuring that the data supplied into our system is valid and that it meets the business rules. We used the following methods to do this:

Datatype Validation:

- Ensuring that dates are entered in valid date/time format that we can work with. Done by using a date picker component.
- Blank field entries are also prevented by using display messages to inform the user about which fields are required to be filled.

Duplicate Entry Checks:

- The system should check through the Reservations database to ensure that no already existing booking can be made more than once for the same period.

Automated Calculations:

- To prevent any potential user error, the system should automatically calculate total costs.

7.4. Output Integrity Controls

Report Controls:

- Accuracy – The system will ensure that booking details are displayed correctly and accurately by checking that the output matches the data stored in the database.
- Consistency – The reports should display data consistently. e.g. Dates should be formatted yyyy/mm/dd for all dates that appear on the report.

  The data shown should also be the most up-to-date entries from the database and any changes to the stored data should be immediately updated in the reports.
- Confirmation – Users will be asked to confirm their inputs at any critical points, for example, when they are about to delete a reservation.

8. Implementation Plan

Scope Statement:

The scope of the project is to design and implement a booking system for the Phumla Kamnandi Hotels. The system must allow a telephone booking to be made, a guest booking to be changed and cancelled and enquiries about guest bookings.

1. Project Initialization (Week 1 & 2)

Objective:

The objective is to define the project scope, determine design requirements for the Phumla Kamnandi Hotel systems and delegate tasks amongst the team.

Tasks:

- Project objectives and boundaries are established, and key deliverables are identified.

- The design requirements are determined, and core functionalities are listed.
- A specification document is drafted that outlines essential features and functionalities.
- Responsibilities are assigned amongst team members for different tasks.
- A timeline is constructed to set project deadlines.

2. System Design & Specification (Week 3 & 4)

Objective:

The objective is to finalize user requirements and technical specifications.

Tasks:

- The system architecture, database structure and frameworks for the backend and frontend are outlined.
- A database schema is developed, and the user interface is designed.

3. Coding & Development (Week 5, 6 & 7)

Objective:

The objective for is to build the system components.

Tasks:

- The database is set up.
- The functionality to create reports is implemented.
- The frontend and backend are linked to allow for data to be retrieved.

4. Testing & Debugging (Week 8 & 9)

Objective:

The objective is to identify errors and bugs within the system and ensure that it is functional and secure.

Tasks:

- Individual components are tested to ensure that they are working correctly in isolation.
- Integration testing is conducted to ensure that different components work well together.
- User acceptance testing is conducted by testing the application from the user’s perspective.
- Debugging is conducted to fix errors in the code.

5.System Delivery & Documentation (Week 10)

Objective:

The objective is to ensure that the system meets the stated requirements, and that all documentation is complete.

Tasks:

- The system documentation is checked to ensure that it is both clear and complete.
- A team meeting is conducted to ensure that the system meets the requirements.

9. Test Plan

The objective of this release is to design and implement a fully functional website that enables customers to book hotel rooms online, make payments, check room availability, and track their reservation status. As well as to add new clients. This will streamline the current booking process, which is cumbersome and time-consuming. We anticipate that this project will enhance customer satisfaction, working in tandem with efforts to strengthen the Phumla Kamnandi Hotels brand.

Test Environment

Hardware Requirements:

- *PC*
  - *Processor of Intel Core i3*
  - *Ram of 4GB*
- *Connection to the Internet*

Software Requirements:

- *An Operating System of windows/Linux/MacOS etc*
- *Database server – SQL server Version 7 or higher*
- *Visual Studio 2022*

Test Items

- *Booking System: Functionality of booking, viewing, editing, and deleting a reservation*
- *User Interface: Clean and simple with no glitchy interface, and a snappy performance*
- *Reporting system: Provides accurate dynamic reports regarding the occupancy*
- *Guest System: Functionality of creating new clients and searching for existing clients*
- *Database: database operations perform correctly without allowing for sql injection attacks*

TEST APPROACHES

We will be using a combination of manual and automated testing for this project.

The approach to test software will be as follows:

1. *Manual Testing:*
   The user interface for the booking process will be tested via manual testing
2. *Security testing:*
   Ensure no SQL injection attacks can occur to corrupt the main database
3. *Integration testing:*
   The system will be tested and monitored to ensure all components work together seamlessly when being used
4. *Unit Testing:*
   Testing individual classes and functions to ensure they work correctly
5. *Usability Testing:*
   Since the objective of this system is to book a room, usability testing will be conducted to ensure that our system is made to be intuitive and user friendly for all users.

PROBLEM TRACKING (TEST CASES)

ID	Test Scenario	Steps to Perform / User action.	Test Data	Expected Results: System behaviour or state.	Result - Comments
1.a.	Guest information validation when creating a	1. A Client attempts to make a booking thru a receptionist, but they are registered so there is no valid customer phone	None	1. The system will output an error message to enter a valid Guest number	

		number to enter for the booking			
1.b.	Room availability Validation	1. Attempt to book a room that's already occupied during the specified dates	1. Select a room that is already booked for the next month	1. System outputs error message that the room is already booked	
1.c.	Date Validation when making a reservation	1. Attempt to reserve a room for a date in the past	1. Enter yesterday's date into the start date for the booking	1. System outputs an error message saying that it is an invalid date	
1.d.	Enter an SQL injection attack in the client's email	1. Attempt to access all data from clients table using an SQL injection	1. "Select * from Clients" inserted into the email when creating a new guest	1. System will treat as normal string and ignore	
1.e.	Client phone number validation	1. Attempt to Enter an invalid phone number when creating a guest	1. Enter an 11-digit phone number	1. System will output error message that the phone number is invalid	
1.f.	Make a booking	1.Attempt to make a booking using valid data	1.Enter appropriate data in all boxes	1.System will output message that the booking has been succesfully added	

TEST SCHEDULE

Milestone	Deliverable	Effort (Person Hour)	Start Date	End Date
Test plan	Detailed test plan for all the features to be developed	15 hours	20/09/2024	23/09/2024
User interface testing	Status of all user interface tests that were conducted and the criteria	5 hours	24/09/2024	25/09/2024
Bug Fixing	Bug status that indicates that all bugs have been fixed and still outstanding bugs	25 hours	26/09/2024	29/09/2024
Operational readiness review	Ensuring all aspects of the system and the status of the system is ready for submission	5 hours	1/10/2024	3/10/2024