

DEPARTMENT OF INFORMATION SYSTEMS  
**SYSTEMS DESIGN & DEVELOPMENT**



## SYSTEMS SPECIFICATION FOR RESTEASY HOTEL SYSTEM

### TEAM MEMBERS

<i>Student Number: ARNSHA011</i>	<i>Student Name: Shai Aarons</i>
<i>Student Number: ORTAND001</i>	<i>Student Name :Andrea Orton</i>

### PLAGIARISM DECLARATION

- We know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.
- This Systems Specification is our own work.
- 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: Shai Aarons

Signed:

Date: 30/09 /2019

Full Name: Andrea Orton

Signed:

Andrea  
Orton

Date: 30/09/2019

## TABLE OF CONTENTS

---

<b>INTRODUCTION</b>	<b>4</b>
OVERVIEW OF SPECIFICATION	4
CONTEXT & SCOPE OF SYSTEM SPECIFICATION	4
DESIGN ASSUMPTIONS & CONSTRAINTS	5
<b>USER INTERFACE &amp; DIALOGUE DESIGN</b>	<b>7</b>
INTERFACE FLOW DIAGRAMS	7
SCREEN STANDARDS	8
DETAILED SCREEN LAYOUT	8
<b>DESIGN SEQUENCE DIAGRAMS</b>	<b>12</b>
DESIGN SEQUENCE DIAGRAM: MAKE A BOOKING	12
DESIGN SEQUENCE DIAGRAM: CANCEL A BOOKING	13
<b>DESIGN CLASS DIAGRAMS</b>	<b>13</b>
<b>ENTITY RELATIONSHIP DIAGRAM</b>	<b>15</b>
<b><i>Report Design</i></b>	<b>16</b>
REPORT 1: OCCUPANCY REPORT	16
<i>Detailed Output Requirements</i>	17
<i>Report Layout</i>	18
REPORT 2: INCOME REPORT	19
<i>Detailed Output Requirements</i>	19
<i>Report Layout</i>	21
<b>INPUT-OUTPUT STANDARDS &amp; CONTROLS</b>	<b>21</b>
FORMALISED OUTPUTS:	21
BUILT-IN VALIDATION TO ENSURE REQUIREMENTS ARE MET	22
INPUT INTEGRITY CONTROLS	22
OUTPUT INTEGRITY CONTROLS	22
<b>IMPLEMENTATION PLAN</b>	<b>23</b>
<b>TEST PLAN</b>	<b>23</b>
TEST ENVIRONMENT	23
TEST ITEMS	24
TEST APPROACHES	24
PROBLEM TRACKING (TEST CASES)	24
TEST SCHEDULE	25



# 1. INTRODUCTION

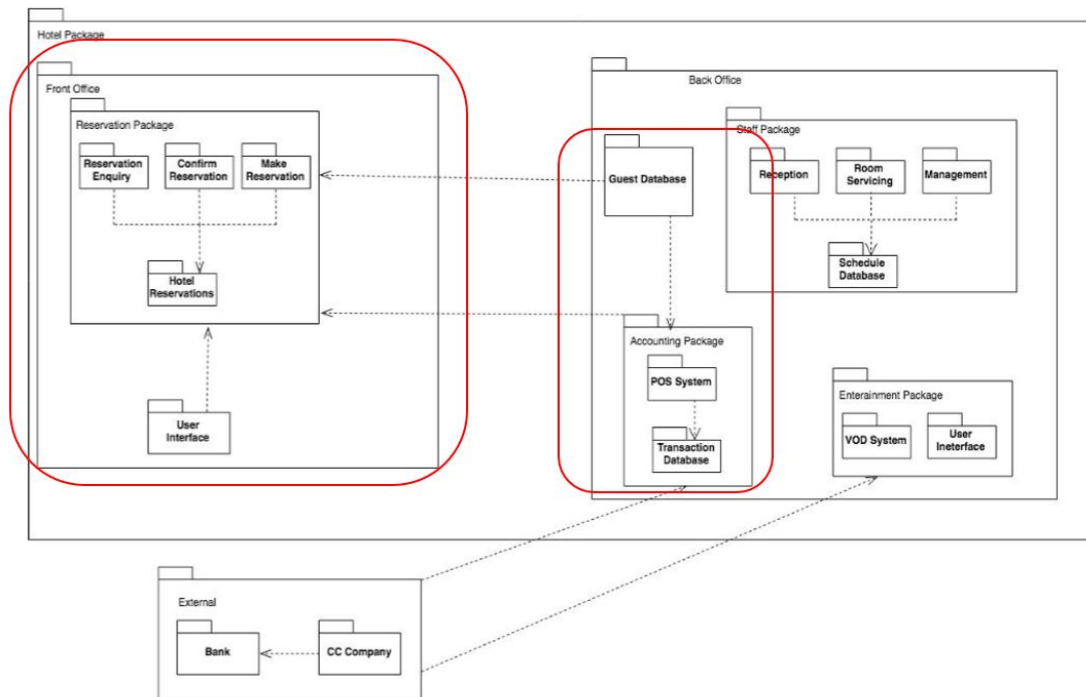
---

## 1.1. OVERVIEW OF SPECIFICATION

This is the Systems Specification Document for the system created for the RestEasy Hotels Franchise. Within this document the User Interface and Dialog Design will be outlined through interface flow diagrams and a detailed layout of the given screen layout for the system. The overall system is outlined within the Design Sequence Diagrams, Entity Relationship Diagram and the Design Class Diagram, followed by relative report designs and Input-Output standards and controls. This document is concluded with a Test Plan - consisting of Test Approaches (environment and items) in order to create multiple test cases. Throughout the creation of this system and its documentation, reference has been made to the relevant Business Case and User Requirements Specification from the previous course (INF2009F). Currently the code is complete for the project and the system is ready to be distributed to the hotel. In further iterations we will add the ability to request a specific room (as there will be variations in room type) and functionality will be added whereby different modules of the hotel such as restaurant, video-on-demand service and bar will be connected to the guests individual accounts.

## 1.2. CONTEXT & SCOPE OF SYSTEM SPECIFICATION

This project aims to create a system for the RestEasy Hotel booking process. The problems that RestEasy has incurred, is a variation in processes within each hotel, with how bookings are created and updated - depending on the software and hardware available in each branch. This system ultimately aims to standardise the booking process across all hotels for RestEasy, to better the overall workings of the business as a whole. RestEasy requires a system that stores relevant details per guest, dependent on their booking, and the associated account per guest (which may be saved for later use, for a returning guest). The system further allows for the RestEasy employee to check availability that is updated daily, depending on the current bookings. The Guest Account stores all payment details and methods for each booking per guest and can be accessed by a RestEasy employee to edit and delete. The system will further include dynamically generated reports that can be accessed by a RestEasy employee - who can select relevant dates for the time period the report will cover. These reports include occupancy levels and income for RestEasy.



It is shown in the diagram above that the Front Office and within the Back Office, Guest Database and the Accounting Package are the chosen areas to be tackled within the system design and development. These areas were chosen due to the booking procedure involving creating a booking (enquiries, confirmation and logging the booking), which is done by a RestEasy employee through the user interface. The process of creating a booking involves the guest database (creating a guest and their account to hold relevant details), which will include the Accounting Package (which is dependent on a guest that is created within the Guest Database). The project deliverables outline all aspects of the process of creating a booking for a RestEasy Hotel room. This consists of requesting a booking, checking availability, creating or accessing a guest and their linked account and further confirming the booking (which updates overall availability within the hotel).

### 1.3. DESIGN ASSUMPTIONS & CONSTRAINTS

Whilst developing the system for the RestEasy Hotel, we made the general assumption that the Hotel gives preference to storage space over performance and speed of the system. This assumption was made due to the fact that the hotel is a business and storing a mass amount of data could lead to extra incurred expenses such as buying more secondary storage, electricity needed for the additional storage and other related costs. Thus we decided to develop the system with storage space in mind and a trade-off was performed with regards to speed and performance of the system.

We saved storage by minimizing the amount of data stored in our databases, coupled with automatic room allocation. To minimize the amount of data we store, we do not store information about

occupancy levels on specific days, but instead store the date that a guest checks in and checks out of the hotel - this way the program can itself devise occupancy and room allocation on its own.

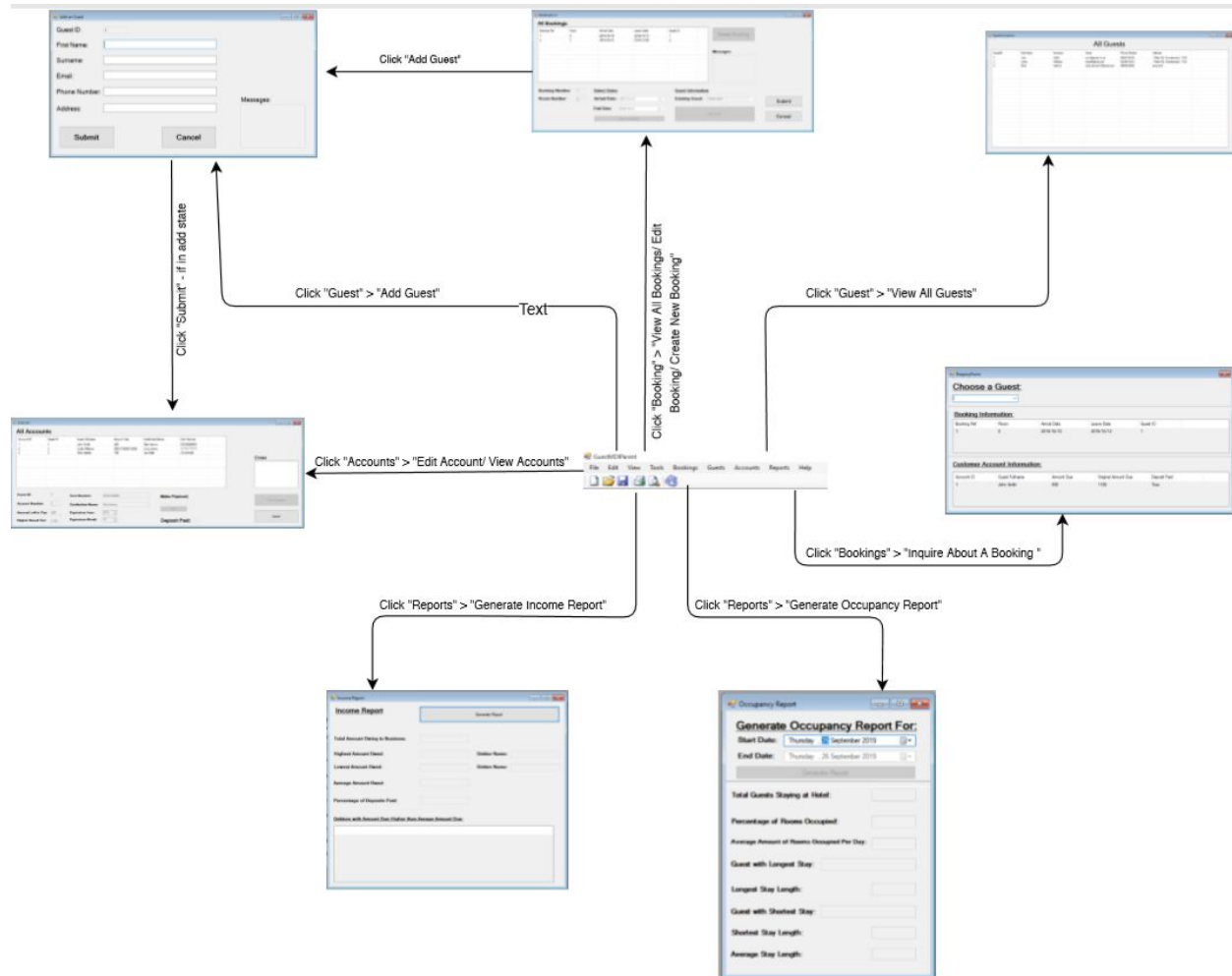
Another storage space optimization that was made, was not storing if a guest has paid his/her deposit or not. Below, it is shown how our program used an algorithm to devise whether a guest has paid his/her deposit without actually having to store if they paid within the database.

Furthermore, an assumption has been made that the rooms should be automatically allocated. The guest does not yet have the ability to request a specific room. We made this assumption due to the fact that all the rooms are of the same size and type - thus requesting specific rooms may not be necessary. However, in further iterations of the system design, this functionality will be added.

The final assumption made is that the system does not export the reports generated to an external file. The reports are generated on-screen within the program and are dynamic in nature. The reason this assumption is made is because the instructions were not clear as to the format and nature of the reports.

## 2. USER INTERFACE & DIALOGUE DESIGN

### a) INTERFACE FLOW DIAGRAMS



The design allows the user to edit, add and delete all information within the same Graphical User Interface - for example (as shown above), the user is able to do as such within the Booking form, and edit and add within the Guest form. The user is also able to add and edit within the same Accounts form.

Utilising the same form for all activities related to the relevant data (being guest data, account data, or booking data), results in a very straightforward and concise system layout, with no room for repetition or changing back and forth between forms, to create the same end result. This speeds up the overall process of creating a booking on the system.

## b) SCREEN STANDARDS

- The design will follow a minimalistic and aesthetic approach. This follows from Jakob Nielson's Usability Heuristics for designing successful user interfaces. A minimalistic and simple design allows for users to make justified decisions and not cloud their judgement by providing an overwhelming amount of options and colors.
- Furthermore, we will be using a simple color scheme of about 3 general colors: A light Grey, faded blue and white color scheme. This color scheme will be applied across all interfaces in the program.
- The layout follows from the general Windows Forms layout - no customized graphics will be implemented as of yet. However, this can happen in further iterations if a graphics designer is added to our team - but for now this is out of scope.
  - The Windows Form tools are already built and designed. They are recognized by the general public and thus provides no confusion about how to use the program.
  - The Windows Form tools are rather simplistic in nature and thus fit in well with our theme of simplicity
- The buttons on a page need to be large and emboldened so that the user knows where to click
- Whenever a control or tool is disabled, it needs to be greyed out to indicate that the user cannot use said functionality

### c) DETAILED SCREEN LAYOUT

[VIEW ALL GUESTS](#)[illegible]



## ADD GUEST

**Add an Guest**

Guest ID:

First Name:

Surname:

Email:

Phone Number:

Address:

**Submit** **Cancel**

**Messages:**

## VIEW ALL BOOKINGS

**BookingForm**

**All Bookings:**

Booking Ref	Room	Arrival Date	Leave Date	Guest ID
1	0	2019/10/10	2019/10/12	1
2	1	2019/12/01	2019/12/05	2

**Delete Booking**

**Messages:**

Booking Number:  **Select Dates**

Room Number:  **Arrival Date:**  **End Date:**

**Guest Information**

**Existing Guest:**  **Submit**

**Cancel**

**New Guest**

**Check Availability**

## CREATE NEW BOOKING

**BookingForm**

**All Bookings:**

Booking Ref	Room	Arrival Date	Leave Date	Guest ID
1	0	2019/10/10	2019/10/12	1
2	1	2019/12/01	2019/12/05	2

**Delete Booking**

**Messages:**

Booking Number:  **Select Dates**

Room Number:  **Arrival Date:**  **End Date:**

**Guest Information**

**Existing Guest:**  **Submit**

**Cancel**

**New Guest**

**Check Availability**

## ENQUIRE ABOUT A BOOKING

**EnquiryForm**

**Choose a Guest:**

**Booking Information:**

Booking Ref	Room	Arrival Date	Leave Date	Guest ID
1	0	2019/10/10	2019/10/12	1

**Customer Account Information:**

Account ID	Guest Fullname	Amount Due	Original Amount Due	Deposit Paid
1	John Smith	400	1100	True

## EDIT BOOKING

**BookingForm**

**All Bookings:**

Booking Ref	Room	Arrival Date	Leave Date	Guest ID
1	0	2019/10/10	2019/10/12	1
2	1	2019/12/01	2019/12/05	2

**Delete Booking**

**Messages:**

**Booking Number:**  **Select Dates**

**Room Number:**  **Arrival Date:**  **End Date:**

**Guest Information**

**Existing Guest:**  **New Guest**

**Check Availability** **Submit** **Cancel**

## VIEW ALL ACCOUNTS/ EDIT ACCOUNT

**Accounts**

**All Accounts**

Account ID	Guest ID	Guest Fullname	Amount Due	Cardholder Name	Card Number
1	1	John Smith	400	Shai Aarons	8888888888
2	2	Linda Williams	500.079986572266	Linda Lewis	7777777777
3	3	Shai Aarons	700	Lexi Katz	3333333333

**Errors**

**Guest ID:**  **Card Number:**  **Make Payment:**

**Account Number:**  **Cardholder Name:**  **PAY**

**Amount Left to Pay:**  **Expiration Year:**  **Deposit Paid:** ☒

**Original Amount Due:**  **Expiration Month:**  **Save Changes** **Cancel**

## GENERATE INCOME REPORT

**Income Report**

Generate Report

Total Amount Owning to Business:

Highest Amount Owed:  Debtor Name:

Lowest Amount Owed:  Debtor Name:

Average Amount Owed:

Percentage of Deposits Paid:

Debtors with Amount Due Higher than Average Amount Due:

## GENERATE OCCUPANCY REPORT

**Occupancy Report**

Generate Occupancy Report For:

Start Date: Thursday, 26 September 2019

End Date: Thursday, 26 September 2019

Generate Report

Total Guests Staying at Hotel:

Percentage of Rooms Occupied:

Average Amount of Rooms Occupied Per Day:

Guest with Longest Stay:

Longest Stay Length:

Guest with Shortest Stay:

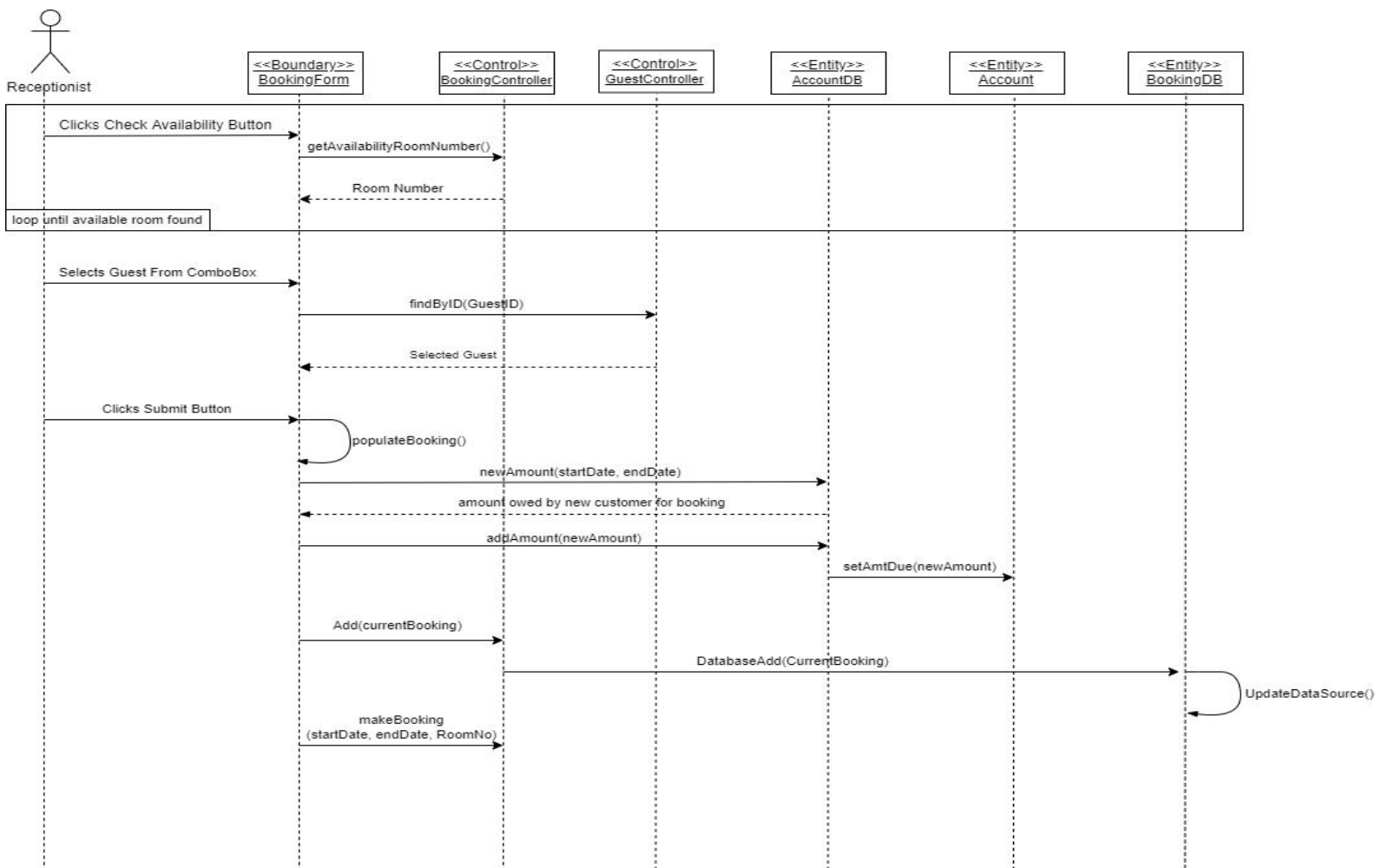
Shortest Stay Length:

Average Stay Length:

### 3) DESIGN SEQUENCE DIAGRAMS

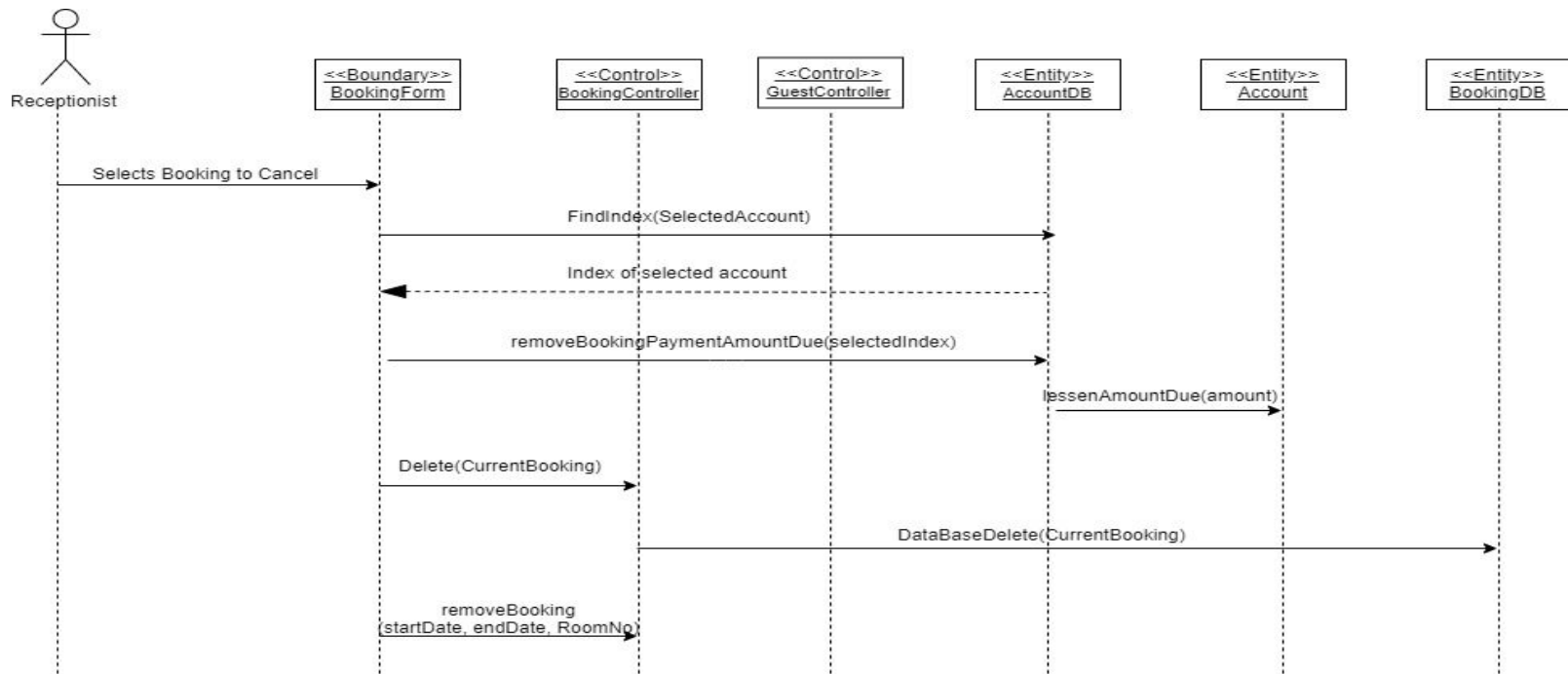
#### a) DESIGN SEQUENCE DIAGRAM: MAKE A BOOKING

- This is the design sequence diagram for the use case: “Make a booking.” This design sequence diagram highlights the process whereby the user creates a booking for a guest to stay at the hotel
- The user starts by selecting dates for the potential booking. Following this the user clicks on a button to check the availability of rooms in the hotel for those dates
- A room number is returned if there is a room available for the selected dates. If the dates are not available, the user searches for dates until a room is available fully for those dates
- A guest is selected from a ComboBox in the boundary and then the “make a booking” process begins

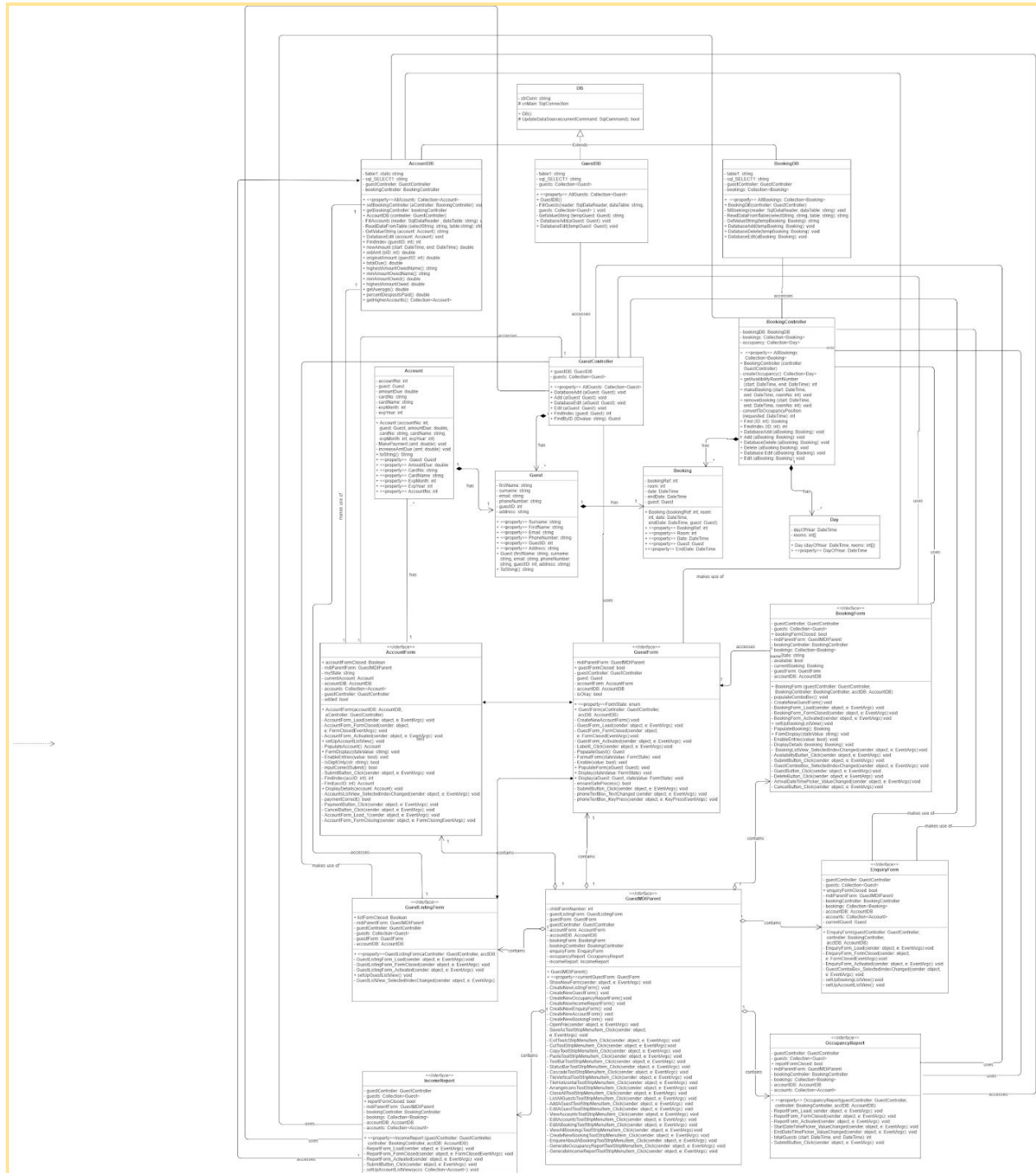


b) **DESIGN SEQUENCE DIAGRAM: CANCEL A BOOKING**

- This is the design sequence diagram that models the process whereby a user wishes to cancel a booking.
- Here the user selects the booking they wish to delete, then the “delete” button is clicked.
- After the delete button is clicked, the cancellation of a booking begins as shown below:

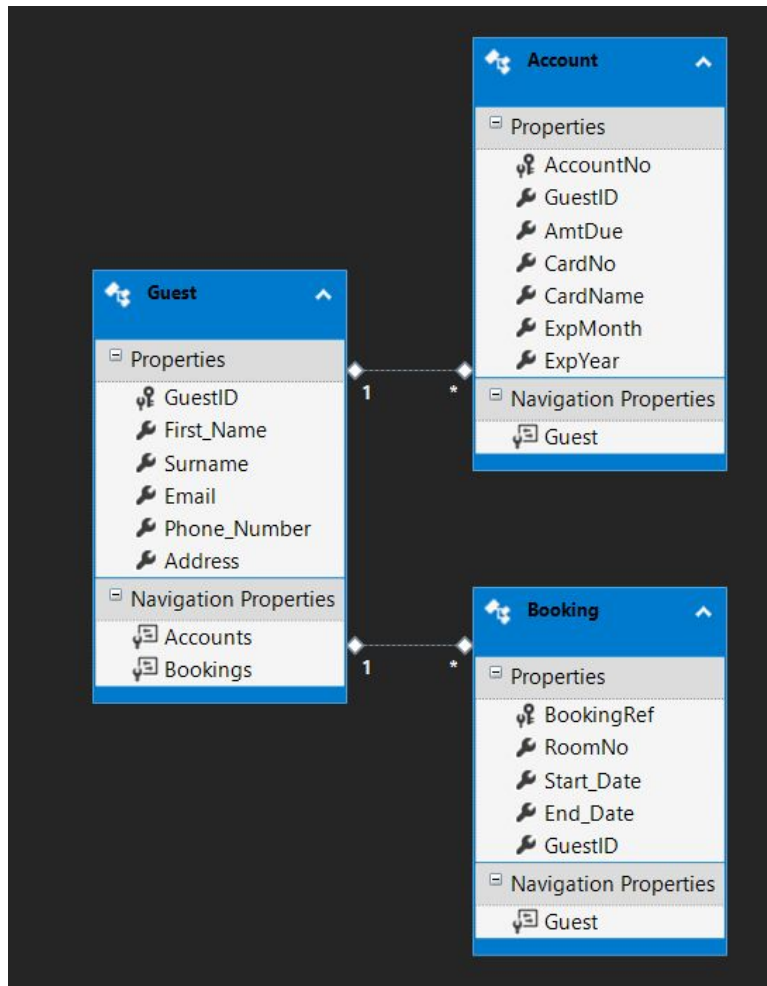


#### 4) DESIGN CLASS DIAGRAMS



## 5) ENTITY RELATIONSHIP DIAGRAM

### ER Diagram



### Data Dictionary

#### Accounts

<u>Key</u>	<u>Name</u>	<u>Data Type</u>	<u>Size</u>
Primary key	AccountNo	int	
Foreign Key	GuestID	int	
	AmtDue	real	
	CardNo	nvarchar	50

	CardName	nvarchar	50
	expMonth	int	
	expYear	int	

**Guests**

<u>Key</u>	<u>Name</u>	<u>Data Type</u>	<u>Size</u>
Primary key	GuestID	int	
	First Name	nvarchar	50
	Surname	nvarchar	50
	Email	nvarchar	100
	Phone Number	nvarchar	50
	Address	nvarchar	150

**Bookings**

<u>Key</u>	<u>Name</u>	<u>Data Type</u>	<u>Size</u>
Primary key	BookingRef	int	
	RoomNo	int	
	Start Date	date	
	End Date	date	
Foreign Key	GuestID	int	

## 6) REPORT DESIGN

---

### a) REPORT 1: OCCUPANCY REPORT

This report is the report that focuses on the occupancy levels of the hotel. This report is dynamic in nature in that a user can select the dates for which this report is relevant for. This report is valuable to multiple stakeholders of the business as it is an overall indicator of how well the hotel is performing and



how full the hotel is over a specific period, chosen by the user. The dates can be adjusted so that different information can be generated depending on the dates that the user selects. Occupancy reports can provide valuable insights for activities such as staff scheduling, accounting and advertising of the hotel.

### ***Detailed Output Requirements***

1. **Output type and ID:** Dynamically Generate Occupancy Report - OccupancyReport
2. **Report Objectives:** An occupancy report to highlight occupancy rates within a specific hotel branch within the RestEasy group for a selected date range. By analyzing this report it will allow management to make decisions in terms of marketing, staff scheduling and for provisional orders.
3. **Audience:**  
 Primary - Hotel Manager of this specific hotel  
 Primary - Receptionist at specific RestEasy hotel branch  
 Secondary - RestEasy executive board  
 Secondary - Marketing Department  
 Tertiary - Finance Department
4. **Content:**  
 In this report, various analyses will be made and shown revolving around the occupancy rates of this specific hotel for the selected dates. For the dates chosen, the report will show information regarding total guests staying at the hotel, the percentage of rooms occupied, the average amount of rooms occupied per day, the guest with longest stay, the guest with the shortest stay and the average amount of days spent at the hotel per guest.
5. **Layout:**  
 This report will follow on from our simplistic and minimalistic design as discussed above. The user will be able to select the dates for the report at the top of the screen and then click a button when he/she is ready for the report to be displayed. The information that comes below the button will follow the simple layout of: Label + textbox. When the button is clicked the textbox's will be populated with the relevant data
6. **Selection:**  
 The data for the occupancy rates will be taken from the database and controllers of the program whereby there is information about who and how long people will be staying at a hotel for.
7. **Sequence:**  
 The information will be shown in a singular vertical manner going downwards
8. **Comparison:**  
 The data will be compared and evaluated using the various controller classes of the program such as the BookingController and the GuestController, as well as the Accounts

table of the database. These items use SQL statements to populate information within the program

**9. Grouping/Summarization**

The grouping on this report is rather simplistic, and there is no necessary importance of placements. The information is provided in a straightforward yet effective manner

**10. Media to be Used**

This is a dynamic, electronic report that can be accessed by any stakeholder using the RestEasy system. By having stakeholders access the report from their individual computers, the company can save paper and extra expenses needed for printing reports.

**11. Frequency, Timing, Delivery**

This report is dynamically generated and thus can be accessed and used at any point during the financial year by any stakeholder who has access to the system. This report can be seen as a simplistic dashboard of sorts, whereby a user can receive information relevant to themselves.

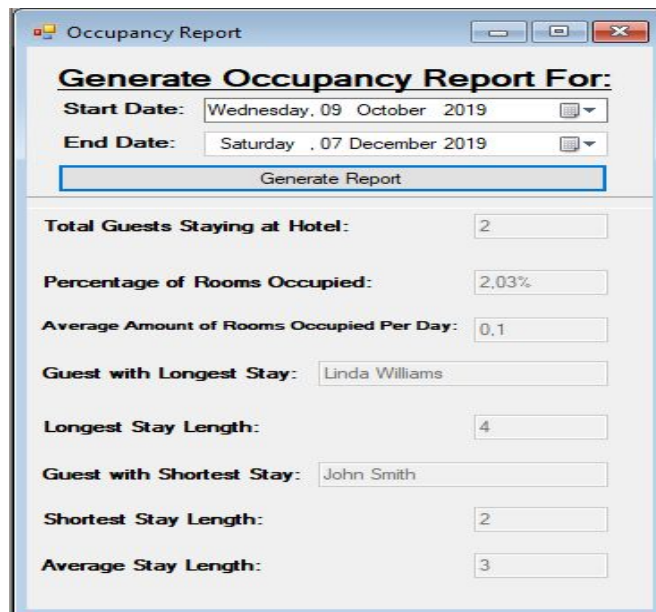
**12. Distribution:**

This report is not distributed, however it is accessed by individual people who have access to the program, who want to draw specific information from the program.

**13. Privacy, Security and Integrity Requirements:**

Since this document will contain sensitive and private information, great care will be taken in terms of not allowing third-party onlookers to be granted information. The report will be password protected so that only the people who are meant to receive this information can read the report - the various stakeholders will be given the password to view the report. This password protection will be implemented in further iterations of the project (see Implementation Plan below)

## ***Report Layout***



**Occupancy Report**

**Generate Occupancy Report For:**

**Start Date:** Wednesday, 09 October 2019

**End Date:** Saturday, 07 December 2019

**Generate Report**

**Total Guests Staying at Hotel:** 2

**Percentage of Rooms Occupied:** 2.03%

**Average Amount of Rooms Occupied Per Day:** 0.1

**Guest with Longest Stay:** Linda Williams

**Longest Stay Length:** 4

**Guest with Shortest Stay:** John Smith

**Shortest Stay Length:** 2

**Average Stay Length:** 3

#### b) **REPORT 2: INCOME REPORT**

This is the report that focuses on the income and financial position of the hotel. This report will focus on the individuals who owe money to the hotel. This report will allow for critical financial analysis and for the financials of the business to be managed in a concise, yet effective manner. This report can be generated at any time of the year, by any person who has access to it - this will allow for instantaneous and immediate financial information to be provided to the hotel stakeholders.

#### **Detailed Output Requirements**

1. **Output type and ID:** Dynamically Generate Income Report - IncomeReport
2. **Report Objectives:** An income report with the purpose of providing financial information to the business and to allow for financial analysis of a specific hotel. This report also provides valuable information about the debtors (people who owe money to the hotel) situation of the hotel. By analyzing this report it will allow management to make decisions in terms of debt management, financial decisions and devising if the hotel is in a decent financial position .
3. **Audience:**
  - Primary - Finance Department
  - Primary - Receptionist at specific RestEasy hotel branch
  - Secondary - RestEasy executive board
  - Secondary - Hotel Manager of this specific hotel
4. **Content:**

In this report, various analyses will be made and shown revolving around the financial information of this specific hotel for the current financial period. The report will show information regarding total amount owing to the business, the debtors with highest and lowest amounts owed to the hotel, the average amount owed to the hotel, the percentage of guests who have paid their deposits and a list of the debtors with an amount owed to the hotel higher than that of the average amount owed to the hotel (Risky debtors).

**5. Layout:**

This report will follow on from our simplistic and minimalistic design as discussed above. The user will be able to click a button when he/she is ready for the report to be displayed. The information that comes below the button will follow the simple layout of: Label + textbox. There is also an area whereby a list of the high-risk debtors are displayed. When the button is clicked the textbox's and the listview will be populated with the relevant data

**6. Selection:**

The data for the income report will be taken from the database and controllers of the program whereby there is each individual guest's account information.

**7. Sequence:**

The information will be shown in a singular vertical manner going downwards, having specific debtor names adjacent to the amount that they owe to the business. The list of high-risk debtors will occupy a large space towards the bottom of the screen.

**8. Comparison:**

The data will be compared and evaluated using the various controller classes of the program such as the BookingController and the GuestController, as well as the Accounts table of the database. These items use SQL statements to populate information within the program

**9. Grouping/Summarization**

The grouping on this report is rather simplistic, and there is no necessary importance of placements. The information is provided in a straightforward yet effective manner. The names of the debtors, with the highest and lowest amounts owed, are placed next to the amount that these debtors owe to the hotel.

**10. Media to be Used**

This is a dynamic, electronic report that can be accessed by any stakeholder using the RestEasy system. By having stakeholders access the report from their individual computers, the company can save paper and extra expenses needed for printing reports.

**11. Frequency, Timing, Delivery**

This report is dynamically generated and thus can be accessed and used at any point during the financial year by any stakeholder who has access to the system. This report can be seen as a simplistic dashboard of sorts, whereby a user can receive information relevant to themselves.

**12. Distribution:**

This report is not distributed, however it is accessed by individual people who have access to the program, who want to draw specific information from the program.

### 13. Privacy, Security and Integrity Requirements:

Since this document will contain sensitive and private information, great care will be taken in terms of not allowing third-party onlookers to be granted information. The report will be password protected so that only the people who are meant to receive this information can read the report - the various stakeholders will be given the password to view the report. This password protection will be implemented in further iterations of the project (see Implementation Plan below)

### Report Layout

Account ID	Guest Fullname	Total Amount Due
3	Shai Aarons	R700

## 7) INPUT-OUTPUT STANDARDS & CONTROLS

### a) FORMALISED OUTPUTS:

When a user performs a use case successfully, this is rather obvious because the entire form transforms and the changes made are instantaneously displayed on listviews. On every user interface form of this project, there will be a message textbox, that will display all relevant information that the user needs to have, such as error messages and room allocation. The outputs generated from each form will be displayed below:

GuestForm:

- "Error: First Name or Surname cannot contain digits or special characters. " - This is displayed when a user inputs a name with numbers or special characters such as '@' or '!'
- "Error: Cannot have empty fields. " - This is displayed when the user leaves a field/textbox empty
- "Error: Invalid Email Address. " - This is displayed when the user inputs an email address in the wrong format. The correct format is 'username@domain'
- "Error: Phone number cannot have letters" - This is displayed when the user inputs a phone number that contains letters

#### AccountForm:

- "Error: Card Number can only have digits. " - This is displayed when a user inputs letters or special characters into the card number field/textbox
- "Error: Card Holder Name can only have letters (no digits or special characters). " - This is displayed when the user inputs a card holder name with letters or special characters
- "Error: Inputs cannot be left empty. " - This is displayed when a user leaves a textbox/field empty

#### BookingForm

- "No rooms available For these dates" - Displayed when the user selects dates whereby the hotel is fully occupied
- "Room number X is available. Please select Guest" - This is displayed when the user requests to make a booking and a room is available for those dates

### b) **BUILT-IN VALIDATION TO ENSURE REQUIREMENTS ARE MET**

In each interface whereby a user can input information there are validation controls to ensure that the input is correctly entered. This allows for the program to function correctly in the case of human error. The controls are shown below for each form where the user inputs information:

- ComboBox's are used for selecting guests in both the BookingsForm and the EnquiryForm - these ComboBox's are populated with all the guest's names so that the user does not input a guest that does not exist in the system
- DateTimePickers are used in both the BookingForm and the OccupancyReport to ensure that the user inputs a date in the correct format
- NumericUpDown's were used for inputting card expiry dates so that the user does not input an invalid expiry date (i.e. month above 12 or below 1)

### c) **INPUT INTEGRITY CONTROLS**

Because our database uses foreign keys our program has to ensure that referential integrity is abided by. This is done through the following procedures:

- Automatically allocating a unique identification number when a new guest, account or booking is created
- Ensuring that the primary keys of each tuple in the program are associated with their respective foreign keys in other tables - refer to ER Diagram above

The program also automatically allocates room numbers when a booking is created so that double-bookings can never occur

#### d) **OUTPUT INTEGRITY CONTROLS**

There are no output integrity controls implemented in the program as this is not necessary. All output is obvious to the user. There are no files exported from the program and thus output integrity controls are not necessary

## 8) **IMPLEMENTATION PLAN**

---

The finalisation of the design of the system will entail adding extra features. This will include:

- A RestEasy employee will have to log in to the system before accessing any company information. The log in will require an individual employee password for security, and each employee will hold different levels of clearance for access to information, dependent on their position in the company.
- Within the booking process, a guest should be able to request a specific room based on their preferences (position in hotel), or by room number. The room numbers are still automatically generated.
- The system should allow for occupancy and financial reports to be able to be exported as PDFs for general use and distribution within the company.
- The system should further allow for changes to the seasonal rates, for yearly shifts in income and type of room available.

Once these standards have been met, the development team will discuss whether these design aspects are attainable, and discuss beginning the creation of these features in a later iteration.

Further approval by the RestEasy stakeholders is necessary in order to start the implementation of the design to the creating and installing of the system. The RestEasy stakeholders will need to verify the following:

- The present design specifications put forward to achieve the given goals of the system.
- The timeline presented for the installation of the new software within all hotel branches.

Once approval has been obtained, the installation process will take place. This involves the standardisation of all hardware within each branch, to ensure the same processes are followed for bookings within each hotel. Following the approval of hardware specifications by RestEasy stakeholders, the software implementation timeline will be put into action. There will be a scheduled approach of installation that involves seminars for current employees to learn how to use the system (dependent on their position) and the installation of the software throughout the company. The installation process aims to be completed within 6 months - with all employees obtaining the correct skills to utilize the software accurately.

## 9) TEST PLAN

---

### a) TEST ENVIRONMENT

Each system needs to have the following minimum requirements to perform testing:

- Sufficient storage space to hold information - recommended 10GB of storage space
- Strong primary storage - Recommend DDR3 8GB of RAM
- Sufficient Central Processing Unit (CPU) - Minimum 4 logical processors at a minimum of i3 or equivalent
- Cloud storage such as Google Drive or Dropbox to hold backup information.
- An Ethernet connection to connect to the server
- Ensure Windows is updated - at least Windows 10 or above
- Ensure that a sufficient anti-virus software is installed - such as McAfee or Avast

### b) TEST ITEMS

We will be testing all the features of this system that can be considered as 'in-scope' for this iteration:

- Testing that a booking can successfully be created and added to the database
- Testing that a booking can successfully be changed and then reflected accordingly on the database
- Testing that a booking can successfully be cancelled and then reflecting this cancellation on the database
- Testing the ability to enquire about an individual's bookings
- Testing the procedure of generating an income report and an occupancy report
- Testing the programs fault tolerance by providing the program with "bad data" with the intention of finding potential issue areas
- Testing that a booking can only be made if a room is available for the selected dates and testing that the automatic room number allocation works.

### c) TEST APPROACHES

The various test methods are to be performed in parallel to the development process. Whenever a new module, method, class or feature is implemented - it needs to be tested by the developer. The program will then go through further testing as described in the test cases below in order to find errors or inefficiencies. The following types of tests are to be conducted:

1. Unit testing - This type of testing is a rigorous inspection of each unit of computation or code. Specific methods and functions of the system will be critically analysed and go through a series of fault-tolerance testing in order to ensure that the tests are passed and any errors are found. This testing occurs on every method and every interface of the program's code
2. System Testing - This type of testing involves evaluating the overall system for completeness and fulfillment of requirements. This type of testing evaluates if the system as a whole fulfills its purpose and required functionality
3. Acceptance Testing - This type of testing can be seen as a demonstration to the potential future users of the program. The stakeholders involved in the RestEasy franchise will evaluate if they accept the system or not.



4. Integration Testing - This will test if the different modules of the system work together in harmony. It tests the success of the integration of the different parts of the system and how efficiently they work together

d) **PROBLEM TRACKING (TEST CASES)**

<u>Test Case Number</u>	<u>Test Case Scenario</u>	<u>Test Data</u>	<u>Expected Response</u>
1	Ensure that rooms are automatically allocated. Make sure that there can never be a double booking	Checking the availability of a potential booking between 25 and 26 December 2019	The message box in the BookingForm should display the following message: "No rooms available For these dates"
2	Ensure that you cannot input any letters into a phone number	Input into textbox: "08289910a8"	The error message box should display the following message: "Error: Phone number cannot have letters"
3	Ensure that a guest can never make a payment above the amount they have due	Select a guest and input into payment checkbox a number higher than that of the amount that the guest is due	The message text box should display an error message of some form
4	Do not allow a new guest to be created without inputting that guests associated account details	Create a new guest with valid parameters	Once the submit button in the guestForm is clicked, the AccountForm will immediately open in the add state, with the GuestID set to that of the newly created guests
5	Ensure that the deposit algorithms correctly ticks the box if the guest has paid the deposit	Create a booking and make a payment of 10% of the booking cost	The deposit textbox should be ticked

e) **TEST SCHEDULE**

<u>Subsystem to be Tested</u>	<u>Duration</u>	<u>Start Date</u>	<u>End Date</u>
<b>Make a booking subsystem</b>	7	2019/09/08	2019/09/10
Unit Testing	5	2019/09/08	2019/09/10
Integration Testing	2	2019/09/08	2019/09/10
<b>Change a booking subsystem</b>	7	2019/09/11	2019/09/13
Unit Testing	5	2019/09/11	2019/09/13
Integration Testing	2	2019/09/11	2019/09/13
<b>Cancel a booking subsystem</b>	4	2019/09/13	2019/09/15
Unit Testing	2	2019/09/13	2019/09/15
Integration Testing	2	2019/09/13	2019/09/15
<b>Enquire about a booking subsystem</b>	5	2019/09/16	2019/09/18
Unit Testing	3	2019/09/16	2019/09/18
Integration Testing	2	2019/09/16	2019/09/18
<b>Report Subsystem</b>	10	2019/09/19	2019/09/22
Unit Testing	7	2019/09/19	2019/09/21
Integration Testing	3	2019/09/20	2019/09/22
<b>Overall system testing</b>	20	2019/09/22	2019/09/29
Integration Testing	10	2019/09/22	2019/09/25
System Testing	7	2019/09/25	2019/09/27
Acceptance Testing	3	2019/09/27	2019/09/29