

ONLINE HOTEL ROOM BOOKING SYSTEM

SOFTWARE ENGINEERING PROJECT REPORT

[Submitted in partial fulfillment]

As a part of the curriculum of
B.SC. (H) COMPUTER SCIENCE



Submitted by:

ISHA RISHI (20044570025)

PRACHI JOSHI (20044570022)

AKRITI RANA (20044570005)

B.Sc. (H) COMPUTER SCIENCE

IV SEMESTER

Mata Sundri College for Women, University of Delhi

Mata Sundri Lane, New Delhi 110002

ACKNOWLEDGMENT

Any project being done always requires the help and support of different people who, through their experience and guidance show us the correct path and keep guiding us with the correct process. Thus, we would like to express our gratitude and we are highly obliged for the help and guidance of our mentor Ms. Ashema Hasti, Assistant Professor, Department of Computer Science, Mata Sundri College for Women, for giving us the right kind of opportunity for the successful completion of our project titled “ONLINE HOTEL ROOM BOOKING SYSTEM”. We thank her for her continuous support, guidance, help, and mentoring before, during the tenure, and after the completion of the project. Her expert guidance and knowledge helped us get through the project successfully. We are also grateful to our friends who helped us in brainstorming sessions to help us analyze in a better way.

Isha Rishi (20044570025)

Prachi Joshi (20044570022)

Akriti Rana (20044570005)

CERTIFICATE BY SUPERVISOR

This is to certify that the project report entitled “Online Hotel Room Booking System” has been submitted by Prachi Joshi, Isha Rishi, and Akriti Rana in partial fulfillment of the requirements of Bachelor of Computer Science (Hons.) embodies the work done by them during semester IV of their course under the supervision of Ms. Ashema Hasti, Assistant Professor, Department of Computer Science, Mata Sundri College for women, University of Delhi.

Ms. Ashema Hasti
(Project Guide)

ABSTRACT

The Online Hotel Room Booking System is a project implemented for “**Dusk Till Dawn Hotel**”, which is an imaginary hotel. It provides people all over the world with an easy and fast way to book hotel rooms online. The interface of the online Hotel Booking System is an App as well as a Web page, that can be accessed with a Web site browser. This app will remove the hectic task of customers and executives in searching and booking rooms. The system will help the administrative staff i.e., executives of the hotel to keep the daily record of the customers in the proper database. Customers can book rooms in advance seating anywhere across the world.

The Online Hotel Booking System is an easy-to-use application. Everyone can easily carry out a booking, modify the booking details, cancel the booking, change and view the booking history, or view the hotel information.

TABLE OF CONTENTS

1. PROBLEM STATEMENT	1
2. SOFTWARE LIFECYCLE MODEL	3
3. REQUIREMENTS ANALYSIS	4
3.1 DFD	4
3.1.1 Context Diagram	4
3.1.2 Level 1 DFD	5
3.1.3 Level 2 DFD	6
3.2 DATA DICTIONARY	11
3.3 USE CASES	
3.3.1.1 Use Case Diagram.....	12
3.3.1.2 Use Case Description.....	12
3.4 SEQUENCE DIAGRAMS	17
4. SOFTWARE REQUIREMENTS SPECIFICATION (SRS)	
4.1 INTRODUCTION	
4.1.1. Purpose	22
4.1.2. Project Scope	22
4.1.3. Overview.....	23
4.2. PROJECT DESCRIPTION	
4.2.1. Product Perspective.....	23
4.2.2. Product Functions.....	23
4.2.3. User Characteristics.....	24
4.2.4. General Constraints.....	24
4.2.5. Assumptions and Dependencies.....	25
4.3. SPECIFIC REQUIREMENTS	
4.3.1. External Interfaces.....	25
4.3.1.1. User Interfaces.....	25
4.3.1.2. Hardware Interfaces.....	26
4.3.1.3. Software Interfaces.....	27
4.3.1.4. Communication Interfaces.....	27
4.3.2. Functional Requirements.....	27
4.3.3. Performance requirements.....	29
4.3.4. Design constraints	
4.3.4.1. Hardware Limitations.....	29
4.3.4.2. Reliability	29

4.3.4.3. Security Requirements.....	29
4.3.5. Software System Attributes.....	30
5. PROJECT PLANNING	
5.1 Project Scheduling	31
5.2 Timeline Chart.....	32
5.3 Effort Estimation & FP –Based Computing.....	33
5.4 Cost Estimation: COCOMO-II Model.....	36
5.5 Risk Analysis.....	38
6. DESIGN	
6.1 Data Design.....	41
7. TESTING.....	43
8. ANNEXURES.....	56
9. REFERENCES.....	63

TABLE OF DIAGRAMS

FIG. NO.	DESCRIPTION	PAGE NO.
2.1	Incremental Model	3
3.1	Context Diagram	4
3.2	Level-1 DFD	5
3.3	Level-2 DFD Registration	6
3.4	Level-2 DFD Login	6
3.5	Level-2 DFD Reset Password	7
3.6	Level-2 DFD Forgot Password	7
3.5	Level-2 DFD Chat Box	8
3.6	Level-2 DFD Feedback	8
3.7	Level-2 DFD Book A Room	9
3.8	Level-2 DFD Booking Cancellation	9
3.9	Level-2 DFD Booking Modification	10
3.10	Level-2 DFD Payment	10
3.13	Use Case Diagram	12
3.14	Sequence Diagram: Register	17
3.15	Sequence Diagram: Login	17
3.16	Sequence Diagram: Reset Password	18
3.17	Sequence Diagram: Chatbox	18
3.18	Sequence Diagram: Feedback	19
3.19	Sequence Diagram: Book A Room	19
3.20	Sequence Diagram: Booking Cancellation	20

3.21	Sequence Diagram: Payment	20
3.22	Sequence Diagram: Booking Modification	21
7.1	Flowgraph	44
7.2	Display UI	55
7.3	Booking History UI	55
7.4	Book A Room UI	55
8.1	Welcome Screen	56
8.2	Registration-1 Screen	56
8.3	Registration-2 Screen	56
8.4	Registration-3 Screen	57
8.5	Login Screen	57
8.6	Reset Password-1 Screen	57
8.7	Reset Password -2 Screen	58
8.8	Forgot Password Screen	58
8.9	Chatbox Screen	58
8.10	Booking Cancellation-1 Screen	59
8.11	Booking Cancellation-2 Screen	59
8.12	Booking Cancellation-3 Screen	59
8.13	Booking Modification Screen	60
8.14	Payment-1 Screen	60
8.15	Payment-2 Screen	60
8.16	Payment-3 Screen	61
8.17	Payment-4 Screen	61

8.18	Payment-5 Screen	61
8.19	Feedback Screen	62

LIST OF TABLES

Table No.	Description	Page No.
3.1	Data Dictionary	11
4.3.1	Hardware Interfaces	26
5.1	Project Scheduling	31
5.2	Timeline Chart	32
5.3	VAF	34
5.4	Weighting Factors of IDVs	35
5.5	Complexity Weights	37
5.6	Productivity Weights	37
6.1	Data Design Table for Customer /Admin	41
6.2	Data Design Table for Feedback	42
6.3	Data Design Table for Bill	42
7.1	Test Cases Table	45

1. PROBLEM STATEMENT

The project is a web-based hotel reservation system for "**Dusk Till Dawn Hotel**" that permits the hotel administrator to address the hotel activities online. Our application provides the leverage and intelligence to regulate the entire procedure from a solitary online system. The hotel reservations project furnishes room booking. The system allows the administrator to dispatch available compartments in the system. Customers can evaluate and reserve a room online. Admin holds the power of either approving or disapproving the customer's recommendation. Hence, the system is beneficial for both customers and administrators to regulate the hotel activities. The existing reservation system adopts a stationary or Microsoft excel spreadsheet and immediate human interaction to book the hotel room and manage reservations. This generates an uncertain exchange of information in the hotel.

Subsequent examination of various traditional hotel room booking systems, we conclude that the major troubles in their system are the following:

- The manual scheme for storing records is not consistent as some inaccuracy can creep in while writing records manually.
- Guests or visitors may face a hard time getting a place to stay in the area. • It is difficult to store records of the availability of the room and the huge number of customers' records.
- More manpower is required and the current system consumes abundant time to make reservations and store data.
- Payment processing and collection are exhausting.
- There is no centralized database that can be created as information, not in one place.
- Stationery is wasted to store the record of available rooms and customers.
- Easily accessible to guest information by unintended users, guest information is extremely unconfident i.e., maintaining file security is difficult.

LIMITATIONS OF THE EXISTING SYSTEM

1. There is an ultimate probability of misplacing customers' records.
2. Maintaining file security and the standard is insufficient.
3. Retrieval of guest records is extremely difficult.
4. Humans are required to fill out forms and enter data, manual data entry leads to errors.

THERE IS TWO ROLE PLAYERS:

- Admin
- Customer

CUSTOMER FUNCTIONALITIES

1. Register (as a customer)
2. Booking Module
3. Login (as a customer)
4. Reset/Forgot password
5. Room Change Request
6. Cancellations
7. Bill generation and payment
8. View Booking history
9. Chat box
10. Submit Feedback

ADMIN FUNCTIONALITIES

1. Login
2. Reset/Forgot password
3. Approve/ Deny Booking request

2. SOFTWARE LIFE CYCLE MODEL

Online Hotel Room Booking System follows **Incremental Process Model**.

We have used the incremental model as it combines elements of linear and parallel process flows. It generates working software quickly and early during the software lifecycle. This model is more flexible and less costly to change scope and requirements. It is easier to test and debug during a smaller iteration. In this model, the customers can respond to each built. Also, functionality can be refined and expanded in the later stages of the later software releases. The user can visualize the software before the completion of the entire project in order to evaluate and provide feedback. We are using this model as requirements are completely understood, however, small changes can be incorporated.

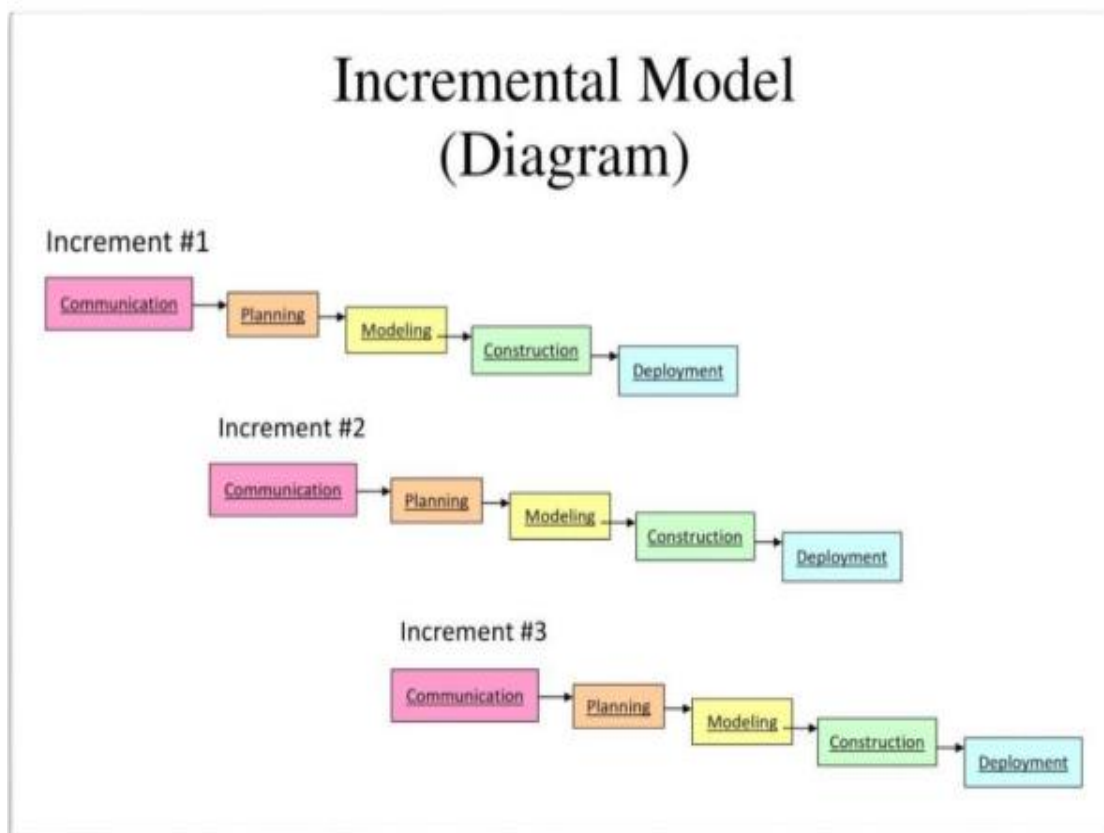


Fig 2.1 Incremental Model

3. REQUIREMENTS ANALYSIS

3.1 DATA FLOW DIAGRAM

3.1.1 CONTEXT DIAGRAM



Fig 3.1 Context Diagram

3.1.2 LEVEL 1 DFD

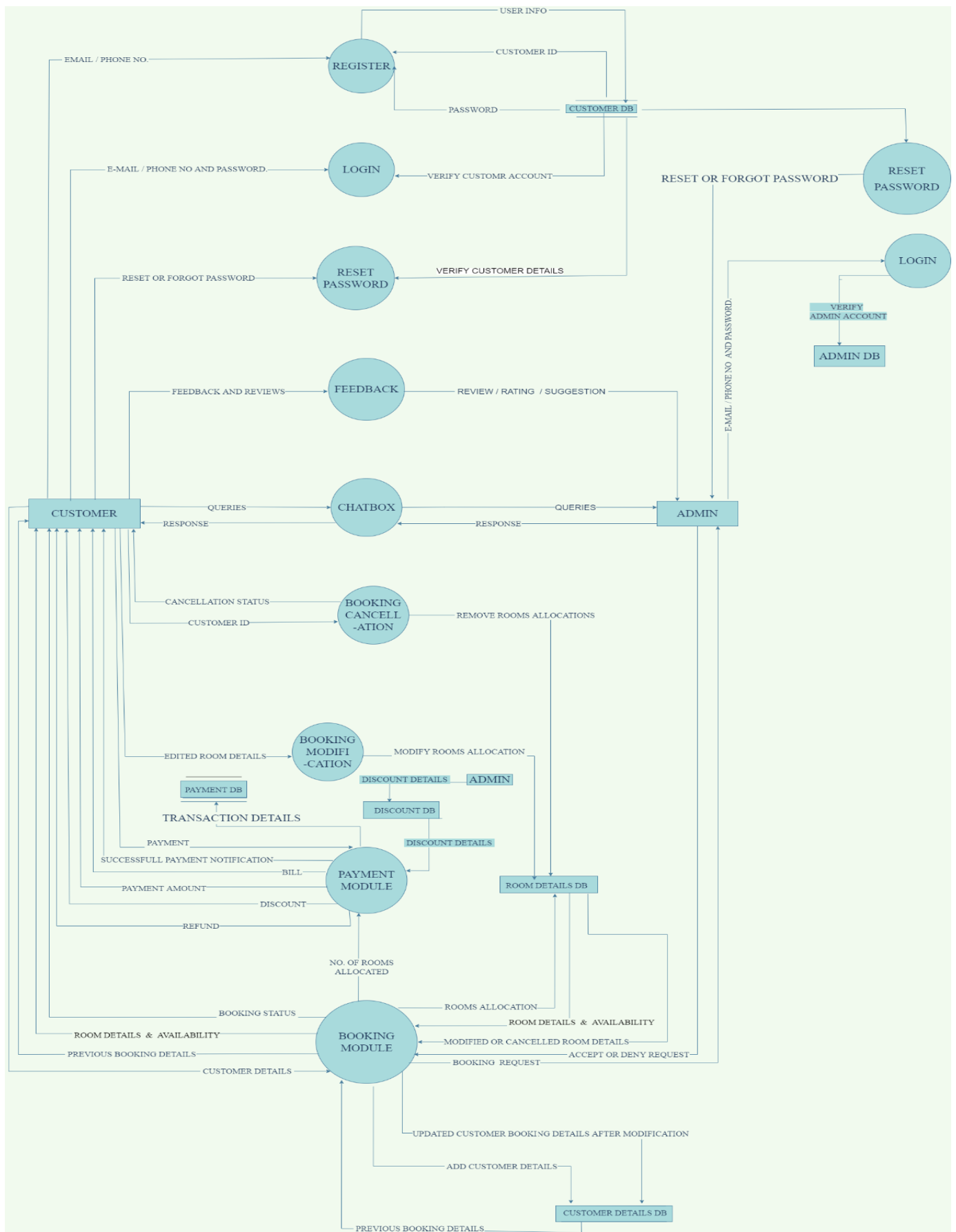


Fig 3.2 LEVEL 1 DFD

3.1.3 LEVEL 2 DFD

REGISTER

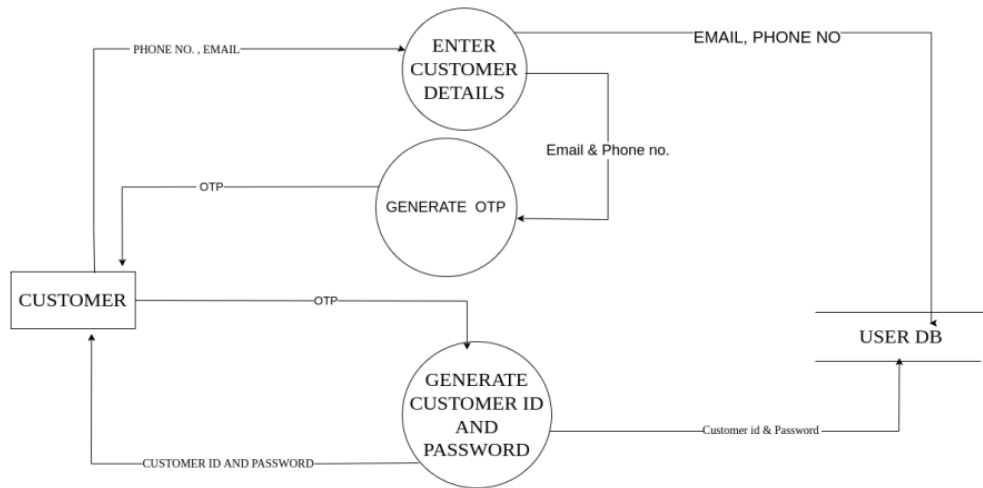


Fig 3.3 Level 2 DFD (Register)

CUSTOMER / ADMIN LOGIN

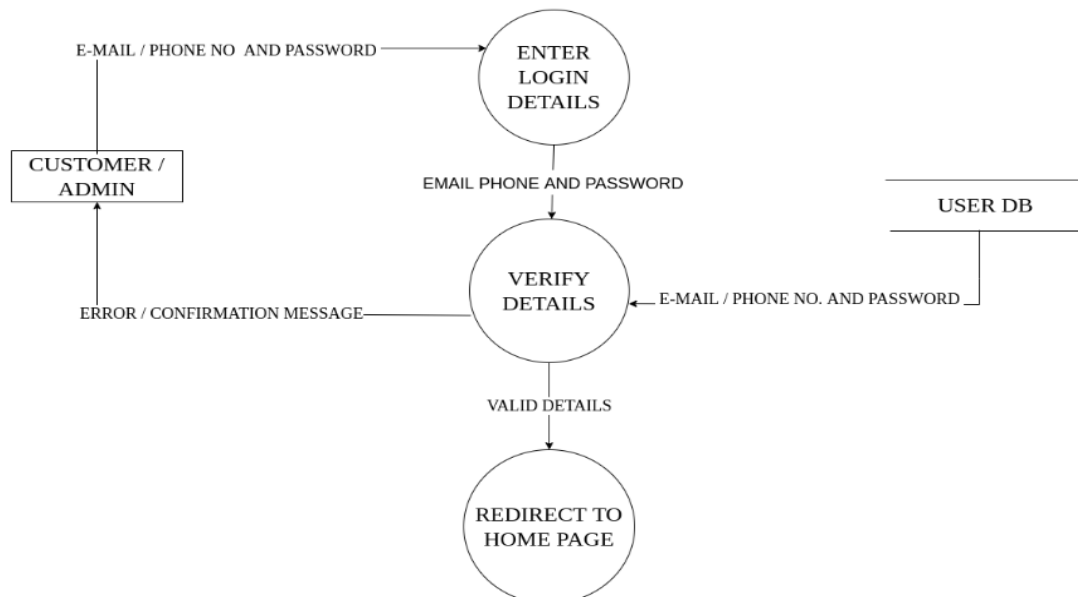


Fig 3.4 Level 2 DFD (Login)

RESET PASSWORD

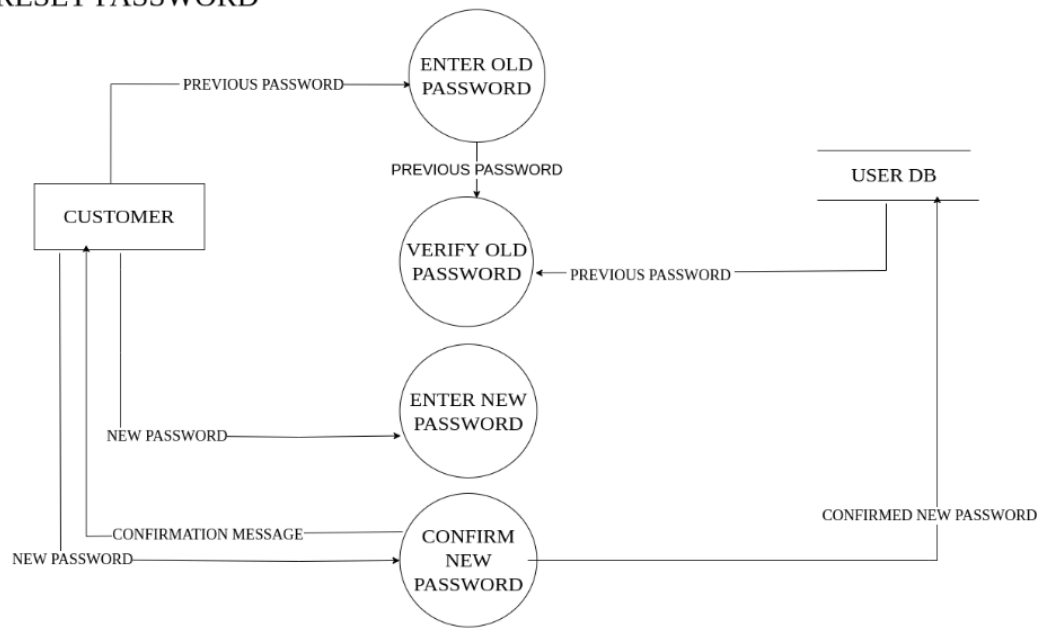


Fig 3.5 Level 2 DFD (Reset Password)

FORGET PASSWORD

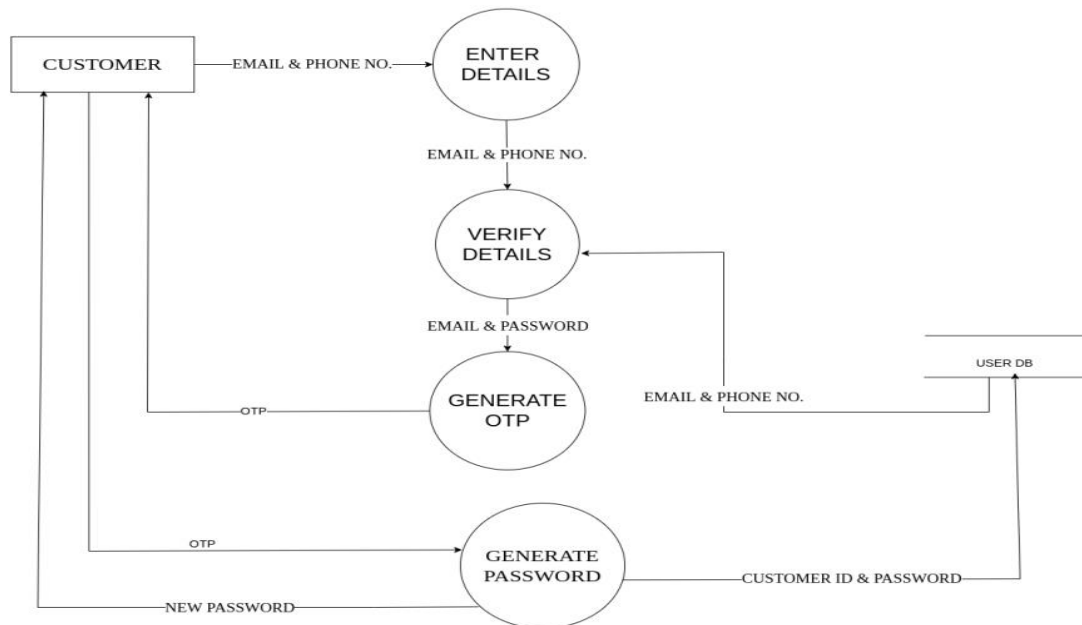


Fig 3.6 Level 2 DFD (Forget Password)

CHATBOX

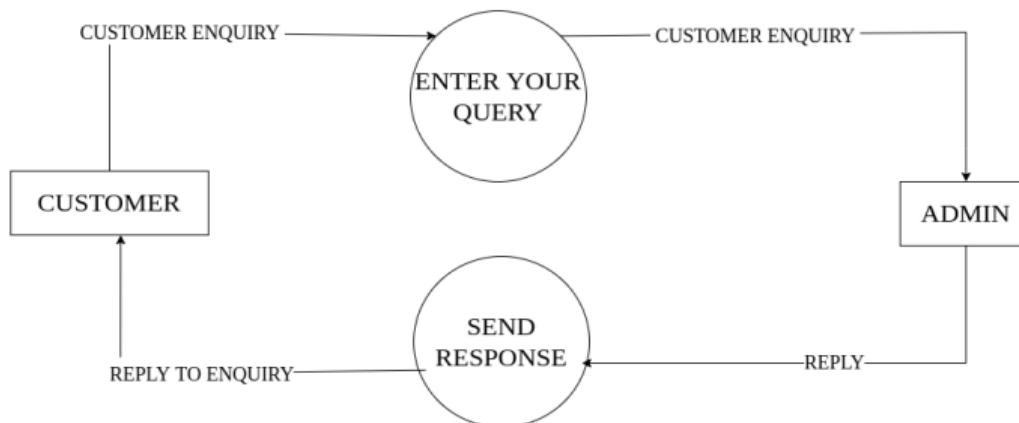


Fig 3.7 Level 2 DFD (Chat Box)

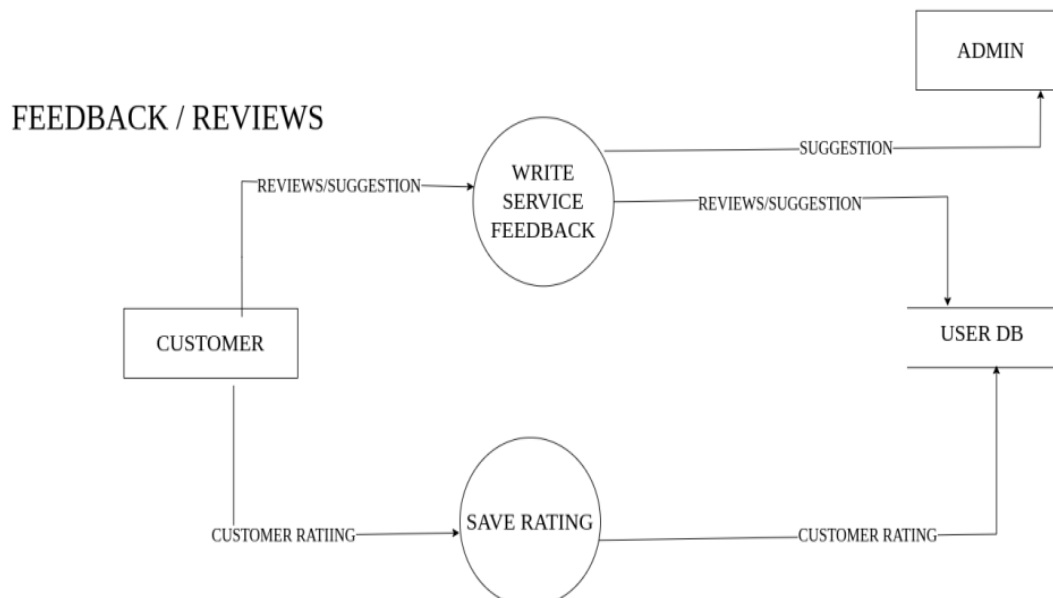


Fig 3.8 Level 2 DFD (Feedback)

BOOK A ROOM

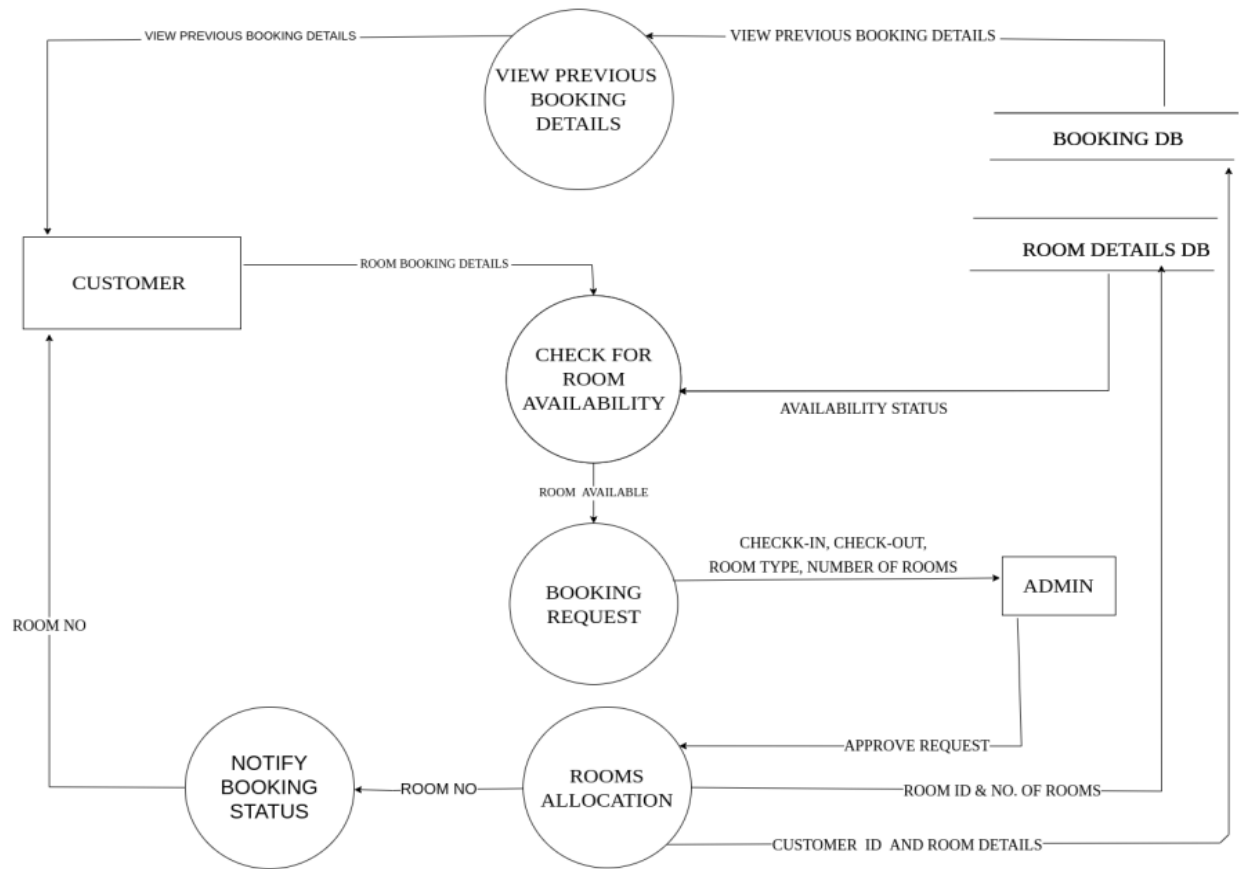


Fig 3.9 Level 2 DFD (Book A Room)

BOOKING CANCELLATION

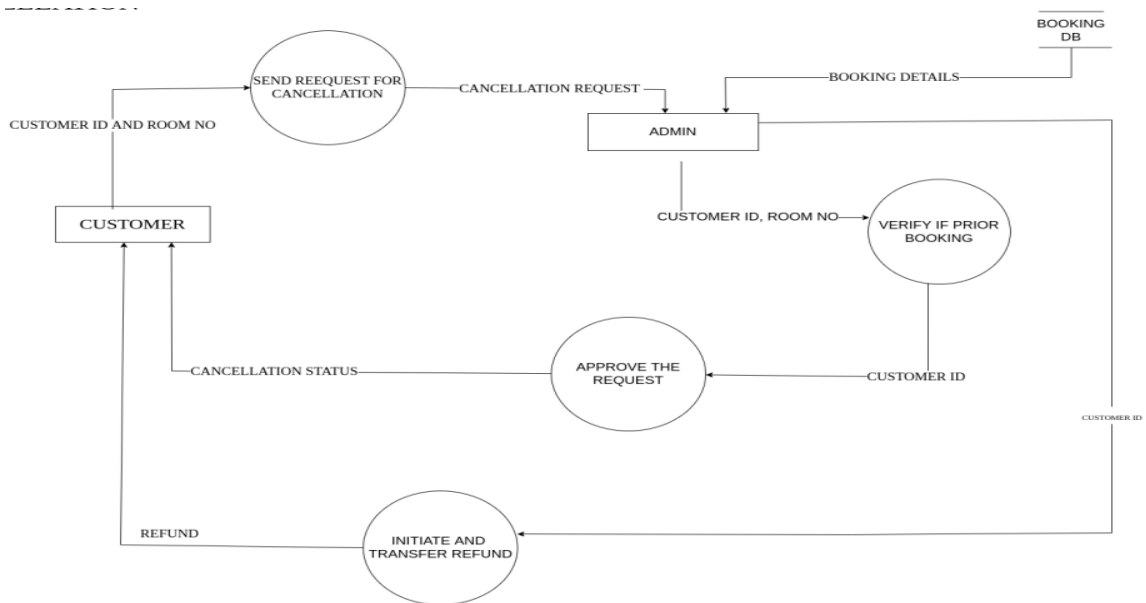


Fig 3.10 Level 2 DFD (Booking Cancellation)

BOOKING MODIFICATION

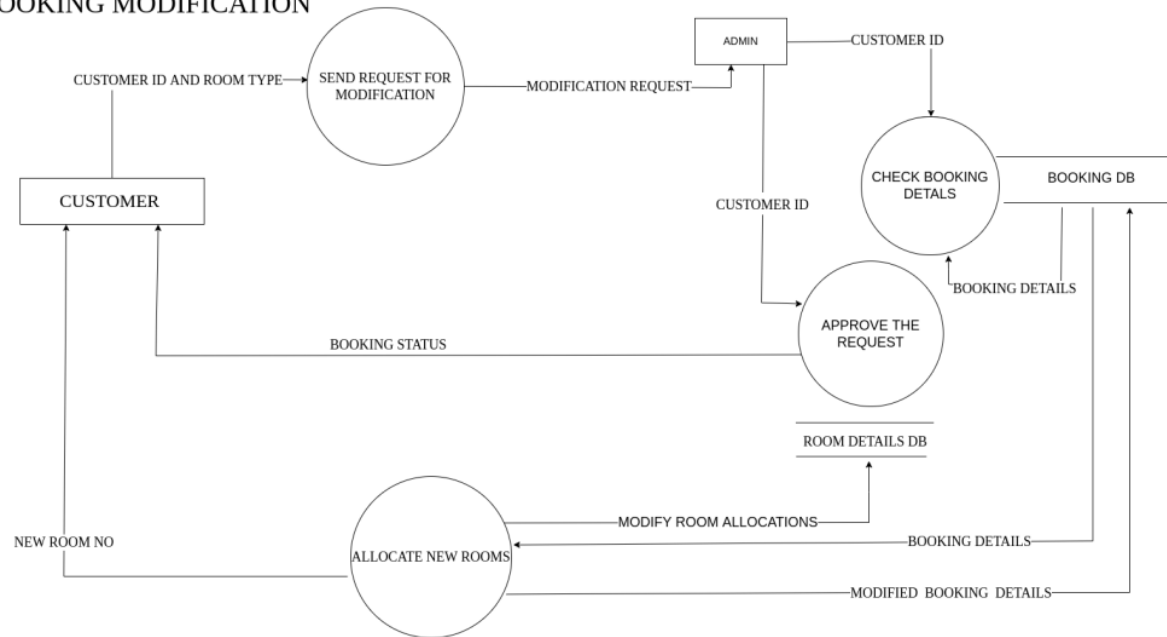


Fig 3.11 Level 2 DFD (Booking Modification)



PAYMENT

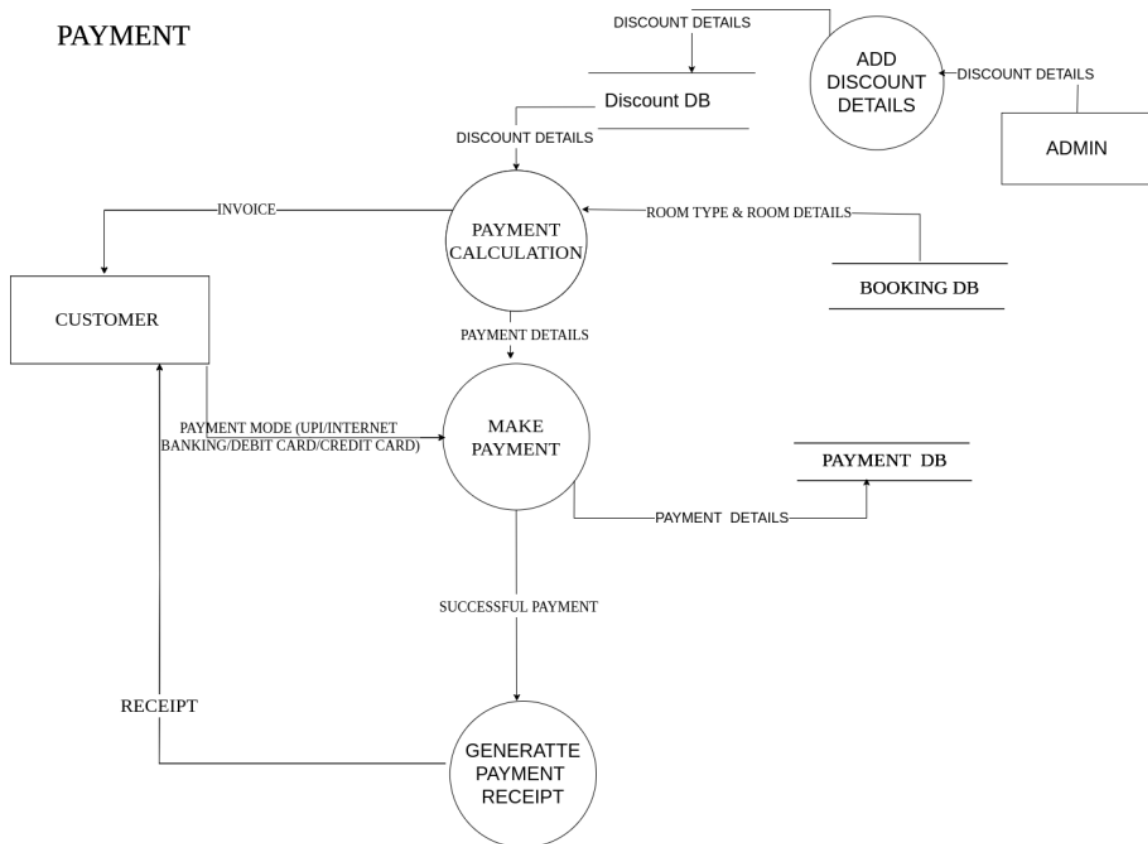


Fig 3.12 Level 2 DFD (Payment)

3.2 DATA DICTIONARY

Legal Characters: [a-z|A-Z]

Digit: [0-9]

Special Characters: [@\$|#|+|-|.]

1	NAME	{Legal character}*
2	E-mail Id	{Legal Characters+Digit+Special Characters}*
3	Username	{Legal Characters+Digit+Special Characters}*
4	Password	{Legal Characters+Digit+Special Characters}*
5	Contact	{Digit+Digit+Digit+Digit+Digit+Digit+ Digit+Digit+Digit+Digit}*
6	Date of Birth	{Digit+Digit+Digit}*
7	Gender	{Legal Character }*
8	Working Hours	{Digit+ Digit }*
9	Prebookig Payment	{Digit+Digit+Digit+Digit} *
10	Offers	{Legal Characters+Digit+Special Characters} *
11	Feedback	{Legal Characters} *

Table 3.1 Data Dictionary

3.3 USE CASES

3.3.1.1 Use Case Diagram

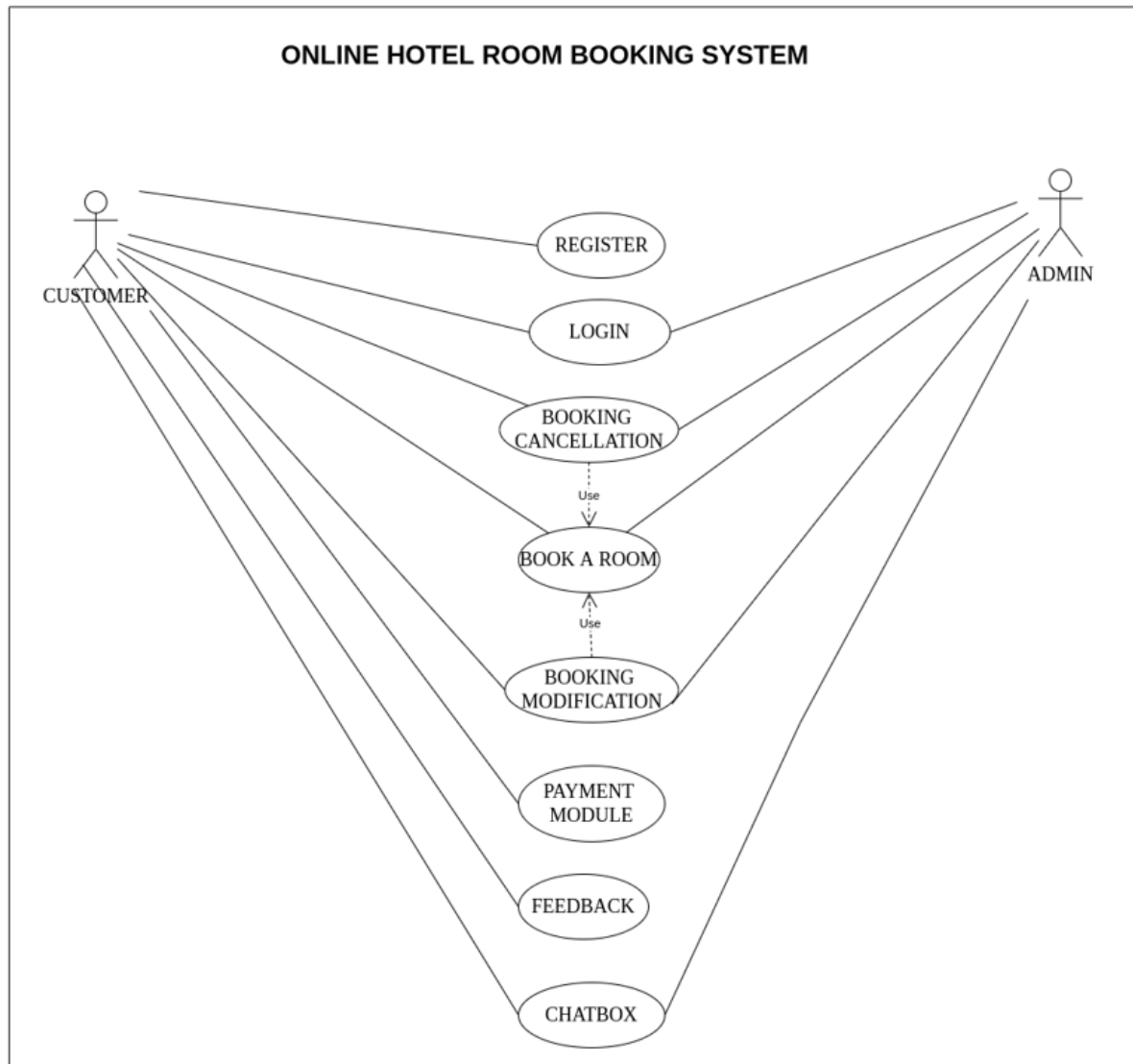


Fig 3.13 Use Case Diagram

3.3.1.2 USE CASE DESCRIPTION

1)REGISTRATION

It allows the actor to register themselves with an email ID and phone number. **ACTORS**

(i) Customer

Flow Of Events:

Basic flow: The actor is required to register him/herself with his/her email ID and password. The system verifies if the ID and phone number are correct or not. After that OTP will be generated, the customer will enter the received OTP and after verification, the customer ID and password will be generated.

Alternative flow: If the customer entered an incorrect email address, OTP, or phone number then it will give an error message.

Pre Conditions: The customer should have a valid email address and phone number

Post Condition: The customer ID and password will be generated if the successful registration has been done.

2) LOGIN

Registered customers will log in to the system using their customer ID and password.

Actors

(i) Admin (ii) Customer

Flow Of Events:

Basic flow: The system requests the actor to enter his/her already registered ID, and password. The system verifies whether the entered ID and the password match the database and allows the actor to log in.

Alternative flow: If the actor enters an invalid Id or password then an error message will be displayed.

Pre-Conditions: The actor must have an already existing account in the system.

Post Condition: If the login is successful, the actor is logged into the system and redirected to the home page. If an actor has the role of a customer, then he/she would have access to the home page. If the actor is an admin, then he would have the customer's information and their booking details.

3) BOOK A ROOM

The customer will book the room using this use case.

Actors

(i) Customer (ii) Admin

Flow of Events:

Basic Flow: The customer will enter the number of rooms, room type and check-in, and check-out date. The system will check the room availability, the room will book respectively and the booking request will be generated which is further accepted or rejected by the admin.

Alternative Flow: If the rooms are not available as per the customer requirement, then the error message will be displayed.

Pre-Conditions: The customer should have logged in to the system.

Post Conditions: After the successful booking the customer will be redirected to the payment module.

Extensions:

(i)Booking Cancellation (ii)Booking Modification

4) BOOKING CANCELLATION

The customer can cancel the booking using this use case.

ACTORS

(i) Customer (ii) Admin

Flow OF The Events:

Basic Flow: The customer can request the cancellation of the booking; admin can accept or deny the cancellation request. Upon the successful booking cancellation, the refund will be transferred.

Alternative Flow: If the admin denies the booking cancellation request the refund will not be generated.

Pre-Condition: The customer should have done a successful booking.

Post Condition: Upon successful execution of the use case, the refund will be transferred.

5) BOOKING MODIFICATION

The Customer can modify the room booking by using this module.

ACTORS

(i) Admin (ii) Customer

Flow Of Events:

Basic Flow: The customer will send the request for the modification of rooms, the admin will process the request. If room modification is possible then admin will accept the request and approve modification. The customer will be allocated the additional rooms as per the request.

Alternative Flow: Admin can reject the booking modification request.

Pre Condition: Successful booking has to be done for further modification.

Post Condition: After successful execution of the use case the additional rooms will be allocated to the customer.

5) PAYMENT

In this use case the customer will make the payment for the booked rooms.

Flow Of Events:

Basic Flow: After booking the rooms, an invoice will be generated and the customer will pay via net banking, credit card, debit card, or UPI, and a payment receipt will be generated.

Alternative Flow: If payment is not done successfully then the booking will be rejected.

Pre Condition: The customer should have done the booking procedure in order to complete the payment.

Post Condition: After payment, a payment receipt will be generated and the customer can visit the hotel as per the check-in date.

6) FEEDBACK

This use case helps customers to give their experience/feedback on the hotel room services.

Actors

(i) Customer

Flow Of Events:

Basic Flow: The customer can give the feedback and rating to the hotel room service and the admin will receive the feedback, also a thank you message will be sent to the customer.

Alternative Flow: None

Pre Condition: Customer should have previous booking details

Post Condition: None

7) CHATBOX

Customers can ask their queries, and the admin will respond to the query accordingly.

Actors

(i) Customer (ii) Admin

Flow Of Events:

Basic Flow: The customer can send the query to the admin, admin will respond to the query.

Alternative flow: None

Pre-Condition: The customer should have logged in.

Post Condition: The customer can ask the query anytime.

3.4 SEQUENCE DIAGRAM

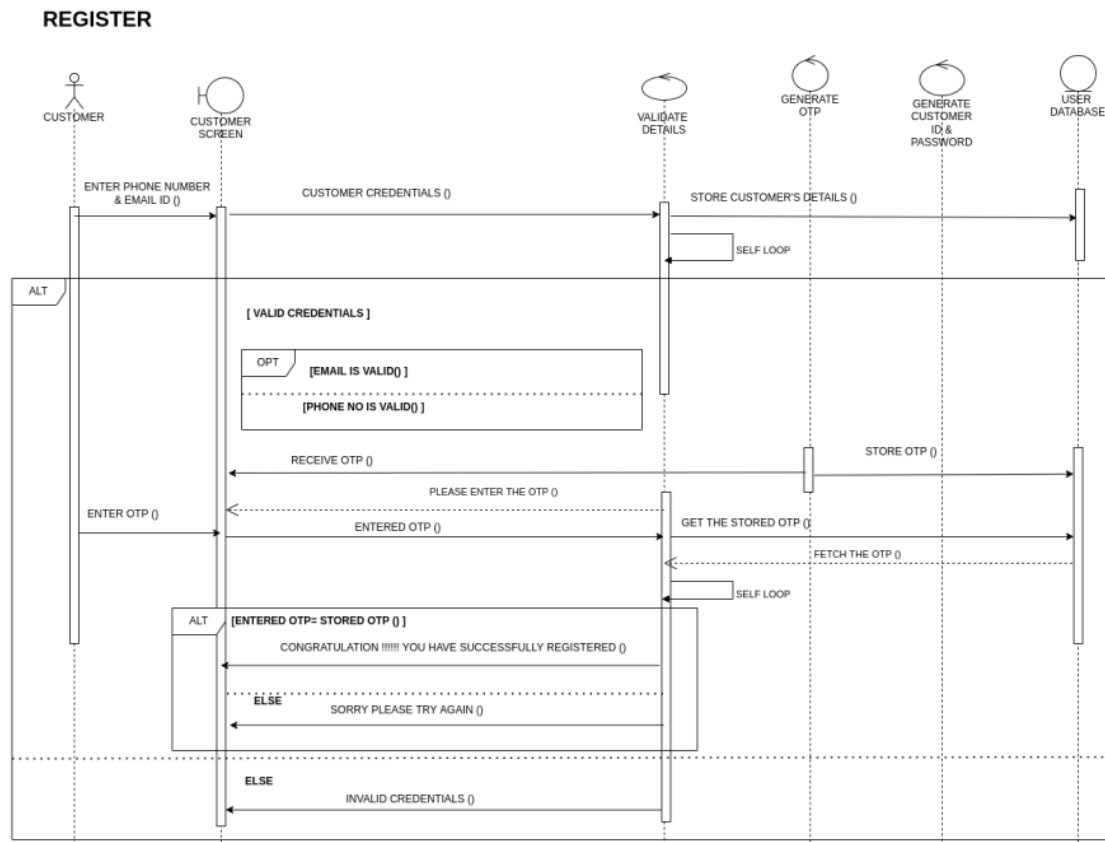


Fig 3.14 SEQUENCE DIAGRAM (Register)

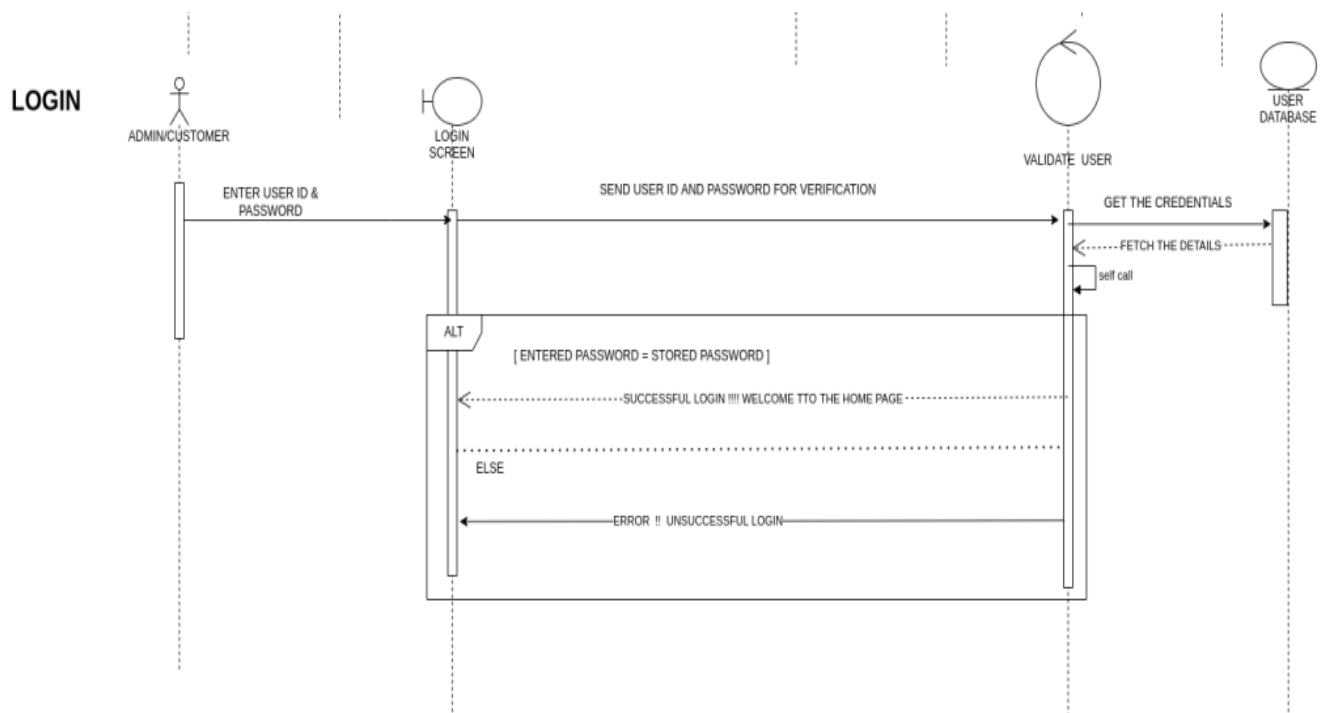


Fig 3.15 SEQUENCE DIAGRAM (Log in)

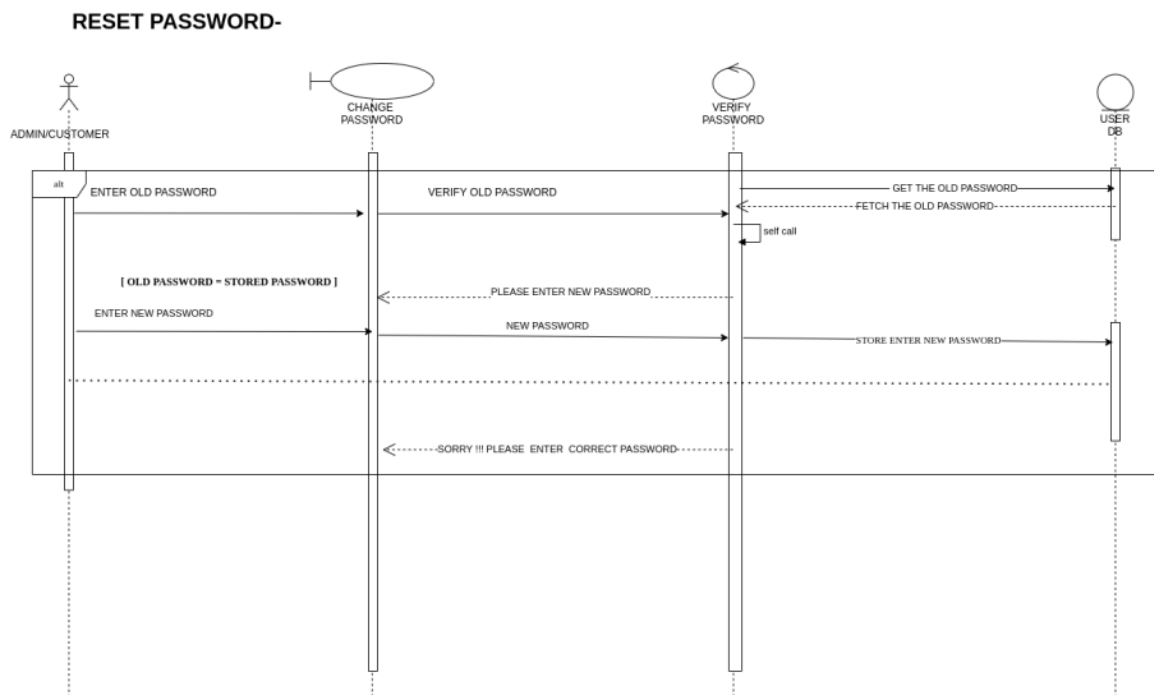


Fig 3.16 SEQUENCE DIAGRAM (Reset Password)

CHATBOX

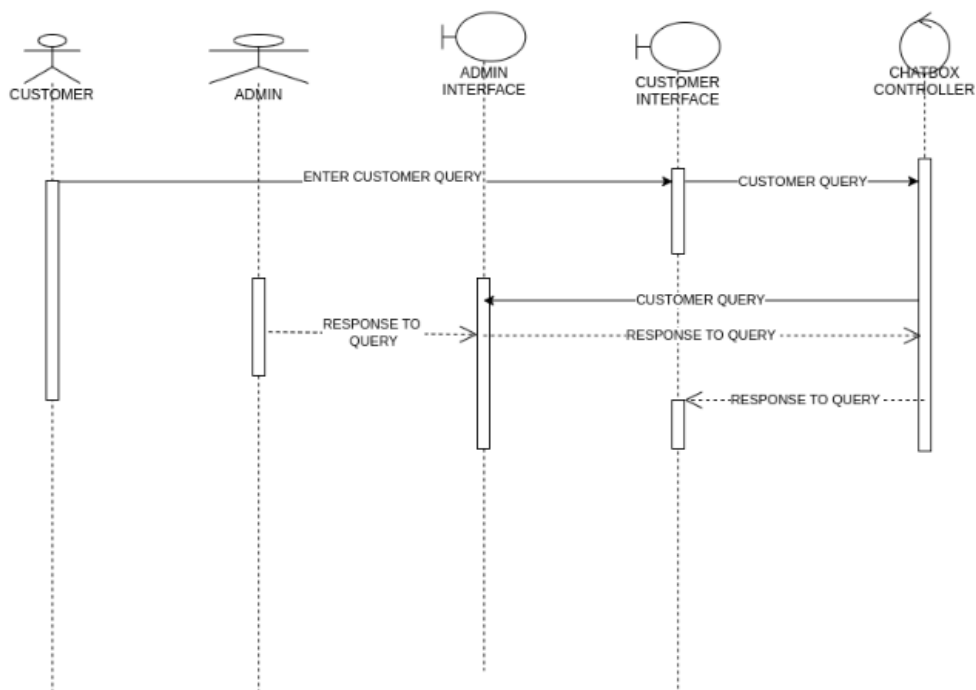


Fig 3.17 SEQUENCE DIAGRAM (Chat Box)

FEEDBACK

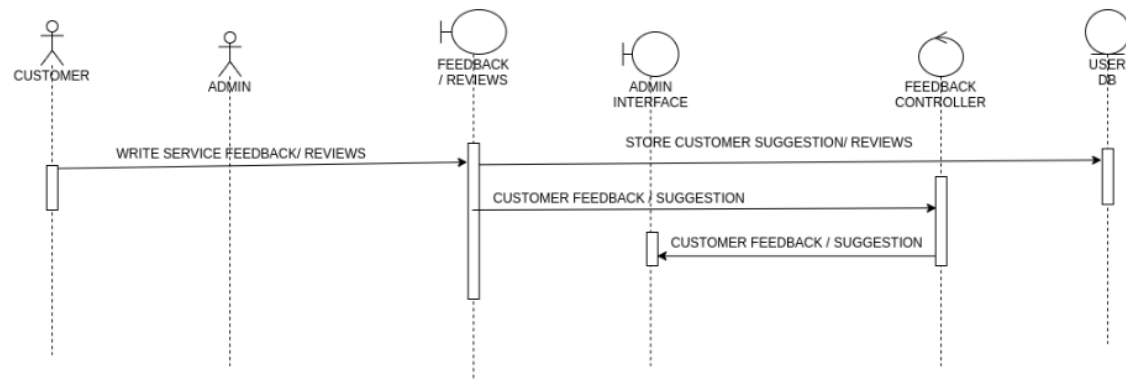


Fig 3.18 SEQUENCE DIAGRAM (Feedback)

BOOK A ROOM

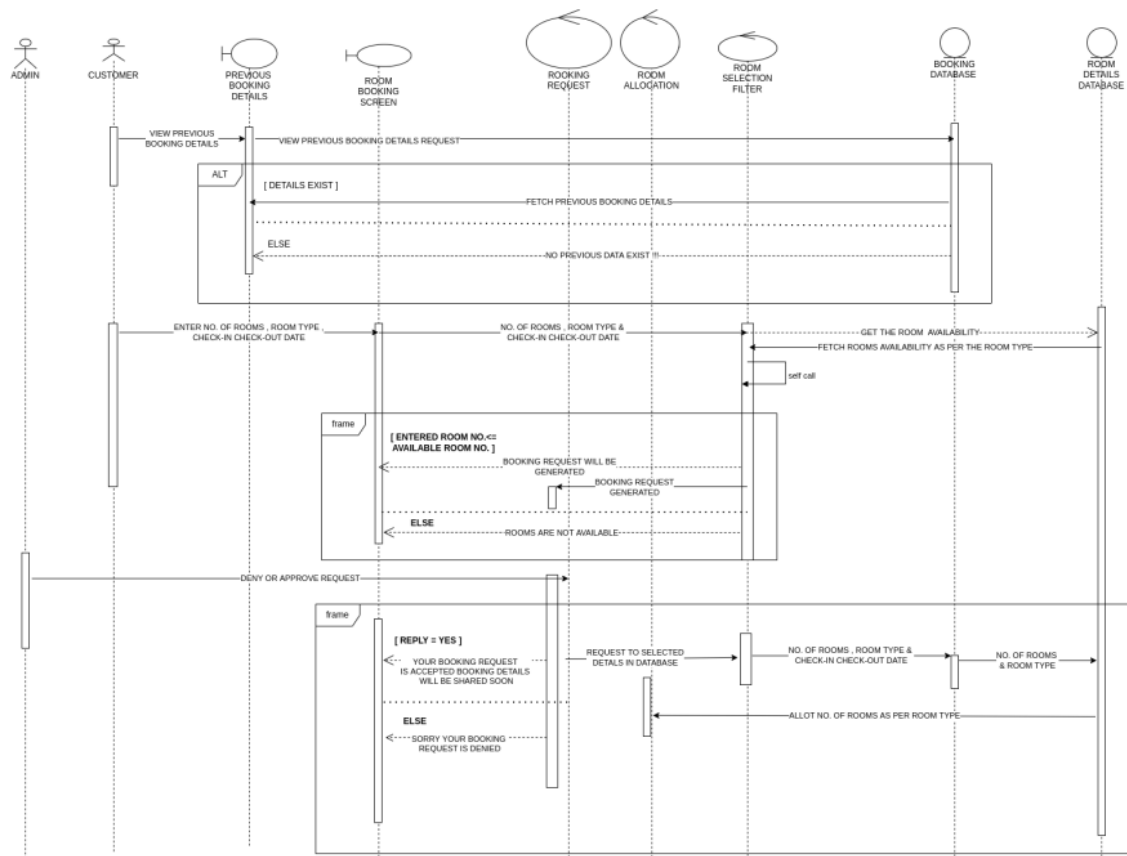


Fig 3.19 SEQUENCE DIAGRAM (Book A Room)

BOOKING CANCELLATION

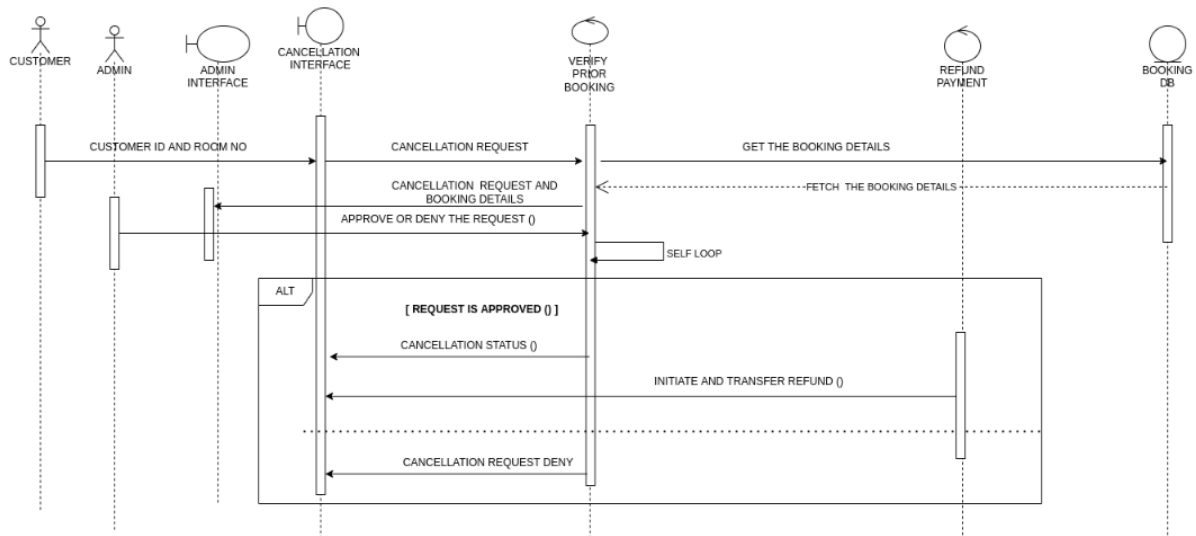


Fig 3.20 SEQUENCE DIAGRAM (Booking Cancellation)

PAYMENT

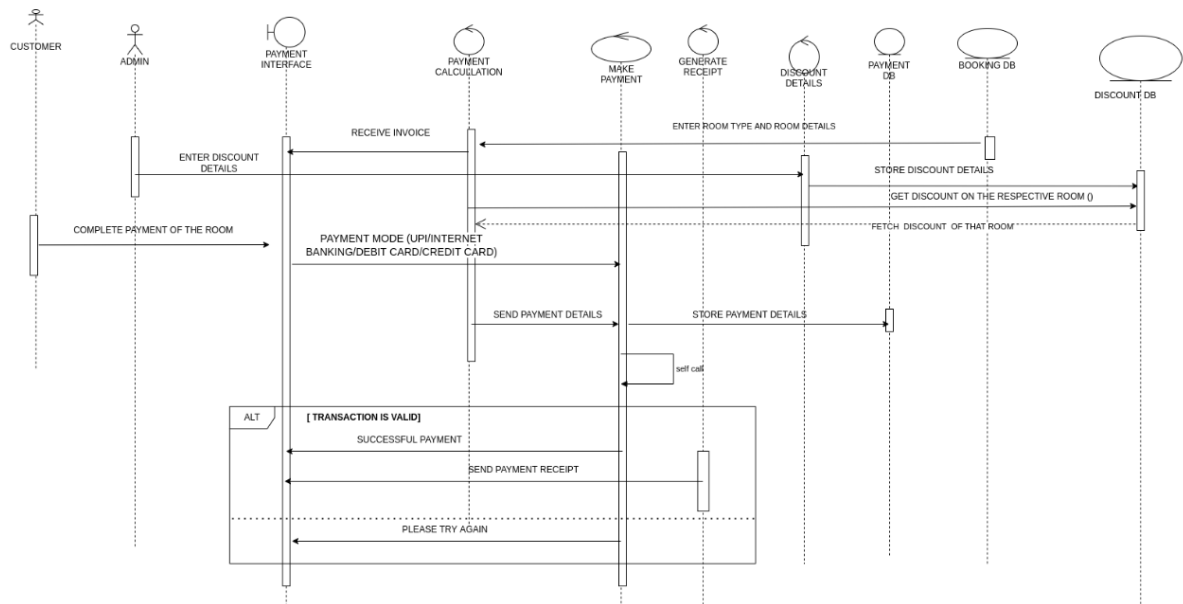


Fig 3.21 SEQUENCE DIAGRAM (Payment)

BOOKING MODIFICATION

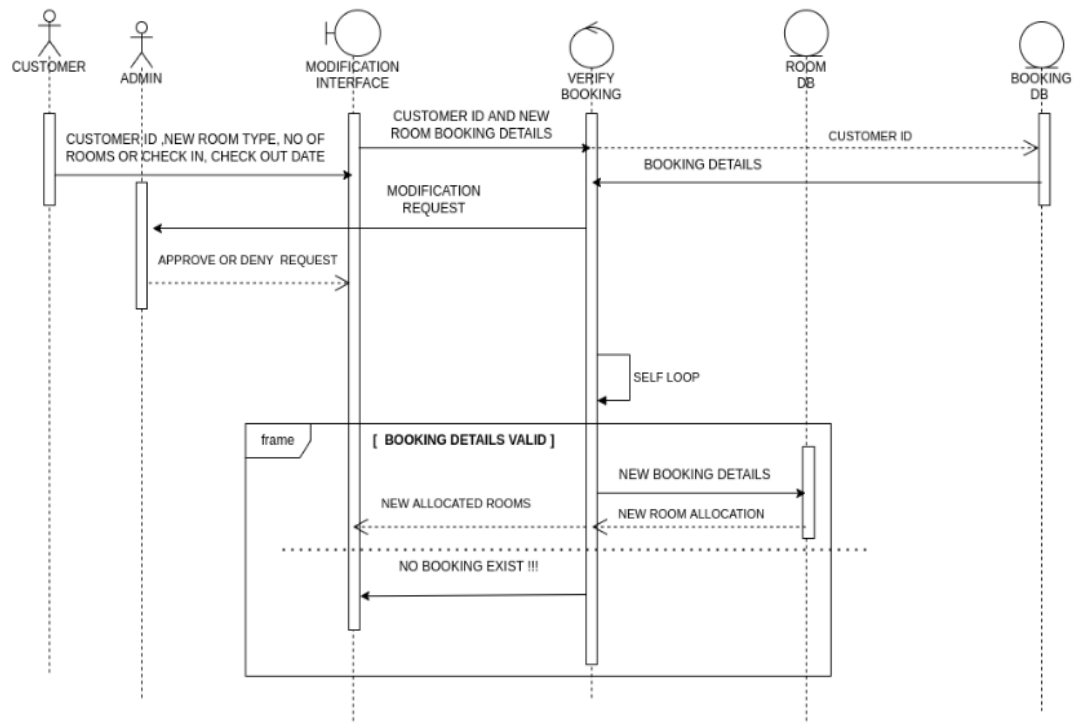


Fig 3.22 SEQUENCE DIAGRAM (Booking Modification)



4. SYSTEM REQUIREMENT SPECIFICATION

4.1 INTRODUCTION

4.1.1 PURPOSE

The Room booking system Software Requirement Specification (SRS) main objective is to provide a base for the foundation of the project. It gives a comprehensive view of how the system is supposed to work and what is to be expected by the end users. Client's expectation and requirements are analyzed to produce specific unambiguous functional and nonfunctional requirements to build a system as per end-user needs.

4.1.2 PROJECT SCOPE

The Room booking system project is intended for the reservation for rooms that is done online. It will be able to do the various operations of Hotel. Our Room booking system will have 2 end users: Customer, Admin. Customers will be able to check room availability, select the rooms, view previous booking history, modify room details, cancellation of rooms, view previous booking history and pay for the rooms. Admin will be able to accept customer request of booking and cancellation.

The main goal of this software is to simplify the everyday process of the hotel. Day to day Hotels is increasing and they need to automate to provide customer ease of access. It will be able to take care of services in a quick manner. This will be able to replace the drawbacks of large customer information physical files which were difficult to handle. Secure Transaction, quick retrieval of information, ease of use, quick recovery of errors, fault tolerance is some of the benefits.

4.1.3 OVERVIEW

The remaining sections of this documentation describes the overall description which includes product perspective and functions, user characteristics, general constraints, assumptions and dependencies. Overall description is listed in section 2. Section 3 includes specific requirements which consists of external interfaces, functional requirements, performance requirements, logical database requirements, design constraints, software system attributes.

4.2 PROJECT DESCRIPTION

4.2.1 PRODUCT PERSPECTIVE

The Room booking system is a software product which will be produced by the team in order to overcome the problems that have occurred due to the current manual system. The system will provide easy access to the system and it will contain user friendly functions with attractive interfaces. The system will give better options for the problems of handling a large scale of physical file system, for the errors occurring in calculations and all the other required tasks that have been specified by the client.

4.2.2 PRODUCT FUNCTIONS

1. Registration
2. Login
3. Book a room
4. Cancellation of rooms
5. Modify Room Details
6. Reset or Forgot Password
7. Make a payment
8. Chat box
9. Feedback

4.2.3 USER CHARACTERISTICS

There are 2 user levels in our Room booking system:

- (i) Customer
- (ii) Admin

CUSTOMER

Customers are a vital part of the system. Customers have access to view room availability and make a payment. They should be able to modify the booking and cancel it if necessary. Customers should at least be capable of using the web UI interface.

ADMIN

Admin has access to accept or deny booking requests, update the discount database and view customers' feedback and reply to chatbox in the hotel system.

4.2.4 GENERAL CONSTRAINTS

Language Requirement: Software must be only in English.

Budget Constraint: The room booking system is intended to be very simple and just for basic functionalities. UI is going to be very simple.

Implementation Constraint: The application should be based on HTML, CSS, PHP, and MYSQL only.

Reliability Requirements: System should sync frequently to the backup server in order to avoid the data loss during failure, so it can be recovered.

Memory: System will need a minimum memory of 512MB. But it is recommended to have a memory of 1 GB.

4.2.5 ASSUMPTIONS AND DEPENDENCIES

It is assumed that the system developed will work perfectly that's going to be developed under the Windows OS. If in case of any difficulties, SRS should be flexible enough to change accordingly.

4.3 SPECIFIC REQUIREMENTS

4.3.1. EXTERNAL INTERFACES

4.3.1.1. USER INTERFACES

Welcome Screen

It shows the welcome page of the Hotel from where customers can register or login. Admin can also login from the same.

Admin Login Screen

Admin needs to enter his/her unique admin id and password to login.

Registration Screen

New customers can register from here by entering email id, phone number. Password and customer id will be generated for the same.

Login Screen

Registered customers can login from here by entering unique customer id and password.

Home Page Screen

After login the customer can view the home page of the Hotel from where he/she can view previous booking history, booking details choose room, make booking, payment, reset or Forgot Password, give feedback and ask queries.

Choose Room Screen

Here the customer can view the room reservation details and other available rooms.

Confirm Booking Screen

After the confirmation of admin, the customer can view the billing details and can proceed to payment.

Payment Screen

Here the customer can view the payment receipt and can do the payment via credit/debit card, mobile banking or internet banking.

Cancellation of Room Screen

To cancel the room the customer has to enter the booking id and room to cancel. Then the cancellation period and refundable amount will be shown to the customer with the cancellation receipt.

Feedback/Rating Screen

Here the customer can give the feedback/rating of booked rooms, hotel staff and services.

Chat box Screen

Here the customer can ask queries or doubts regarding room booking and hotel and admin will give reply to them.

Reset/Forgot Password Screen

Here the customer can reset his/her password or generate new password if forgot.

Modification Screen

Here the customer can modify or update his/her room booking details which is approved by admin.

4.3.1.2 HARDWARE INTERFACES

Server Side				
Monitor	Processor		RAM	Disk Space

Resolution: 1024*768	Intel or AMD 2GHZ		4GB	10GB
Client Side				
Monitor	Processor		RAM	Disk Space
Resolution: 1024*768	Intel or AMD 1GHZ		512MB	2GB

Table 4.3.1 HARDWARE INTERFACES

Reservation alerts will be sent as an e-mail notification. So, there is a need for broadband internet connection. Clients should be able to keep a stable internet connection. A printer will be needed when printing bills.

4.3.1.3. SOFTWARE INTERFACES

The computer on this software is going to be installed and need to have Windows Operating System equal or above, Windows 7.

4.3.1.4. COMMUNICATION INTERFACES

When a specific reservation is reserved, at the same time an email notification will be sent to both admin and customer's email account. To achieve that functionality, it requires having a stable internet connection. Mostly a broadband connection with the customer's computer will provide an efficient service. The System shall be using HTTP for communication over Internet and for intranet communication, it shall use TCP/IP protocol.

4.3.2 FUNCTIONAL REQUIREMENTS

REGISTRATION

The Customer should be able to register with their details.

The system should record following customer details into Customer Database.

Phone no

Email ID

The system shall send verification message to the customer's email.

LOGIN

The system should verify the customer id & password against the member database when logging in.

After login, customers should be directed to Home Screen.

CHOOSE ROOM TYPE

The customer inputs filters such as time period, check-in, check-out.

It filters out the available room details.

BOOK A ROOM

The customer input required details to no. of rooms , check-in date, check-out date.

The system validates the entered details, after that booking request is further generated and approved by admin and records the information in the booking database and room details database.

MAKE PAYMENT

The system should allow customers to pay bills online via mobile banking, internet banking, and debit/credit card.

MODIFY BOOKING DETAILS

The system should allow customers to modify their booking details as per requirement and the system will generate a modification request for

the same and the admin will approve or deny the request, after that the modification will be done successfully.

4.3.3. PERFORMANCE REQUIREMENTS

Performance requirements define acceptable response time for system functionality. The performance of the system will highly depend on the performance of the hardware and software components of the installing computer. When considering the timing relationships of the system the load time for user interface screens shall take no longer than two seconds. It makes fast access to system functions. The login information shall be verified within five seconds. Data in the database should be updated within 2 seconds.

4.3.4. DESIGN CONSTRAINTS

4.3.4.1 HARDWARE LIMITATIONS

When the server can handle more number of users, it takes time to process.

4.3.4.2 RELIABILITY

The software is of no use if it doesn't provide reliability. System should sync frequently to the backup server in order to avoid data loss during failure, so it can be recovered.

4.3.4.3 SECURITY REQUIREMENTS

All external communication between the data's server and customer must be encrypted. Payment process should use HTTP to secure the payment transactions. The software should not cause the spread of a virus.

4.3.5 SOFTWARE SYSTEM ATTRIBUTES

Correctness: This system should satisfy the regular Hotel Management operations to fulfil customer requirements.

Efficiency: Enough resources will be there to achieve the task efficiently.

Flexibility: System should be flexible enough to provide space to add new features and to handle them conveniently.

Integrity: System should focus on securing the customer information and avoid data losses as much as possible.

Portability: The system should run in any Microsoft windows environment.

Usability: The system should provide a user manual for every level of users. The system should be user-friendly.

Testability: The system should be able to be tested to confirm the performance and to ensure it performs as intended.

Maintainability: The system should be maintainable.

Availability: The system shall be available during normal hotel operating hours.

Robustness: Strength of the system to handle system functions accurately and maintain the database without facing unexpected failures.

5. PROJECT PLANNING

5.1 PROJECT SCHEDULING

Work Tasks	Planned Start	Actual Start	Planned Complete	Actual Complete	Assigned Person(s)	Effort Allocated
Problem Statement	Jan.w1, d1	Jan.w1, d3	Jan.w2, d3	Jan.w2, d5	Isha, Prachi, Akriti	3 person per week
Software Lifecycle Model	Jan.w1, d2	Jan.w1, d4	Jan.w1, d3	Jan.w1, d5	Isha	1 person per week
Project Scheduling	Jan.w3, d3	Jan.w3, d3	Jan.w3, d4	Jan.w3, d4	Prachi, Isha	2 person per week
Timeline Chart	Jan.w3, d3	Jan.w3, d3	Jan.w3, d4	Jan.w3, d4	Akriti	1 person per week
Software Requirement Specification	Jan.w3, d4	Jan.w3, d5	Feb.w1, d1	Feb.w1, d5	Isha, Akriti, Prachi	3 person per week
Context Level Diagram	Feb.w1, d2	Feb.w1, d2	Feb.w1, d5	Feb.w1, d5	Isha, Akriti	2 person per week
Data Flow Diagram 1	Feb.w2, d1	Feb.w2, d1	Feb.w2, d4	Feb.w2, d5	Isha, Akriti, Prachi	3 person per week
Data Flow Diagram 2	Feb.w2, d3	Feb.w2, d5	Feb.w4, d3	Feb.w4, d3	Isha, Prachi	2 person per week
Data Dictionary	Feb.w3, d3	Feb.w4, d3	Feb.w3, d4	Feb.w4, d5	Akriti	1 person per week
Project Metrics	Mar.w1, d1	Mar.w1, d3	Mar.w2, d1	Mar.w2, d5	Isha	1 person per week
Effort Estimation (Cocomo II model)	Mar.w2, d1	Mar.w3, d1	Mar.w3, d1	Mar.w3, d5	Isha, Akriti	2 person per week
Risk Analysis	Mar.w2, d3	Mar.w2, d5	Mar.w3, d5	Mar.w3, d5	Prachi	1 person per week
Data Design	Mar.w3, d2	Mar.w3, d5	Mar.w3, d4	Mar.w4, d1	Prachi	1 person

						per week
System Design	Mar.w3, d4	Mar.w4, d1	Mar.w4, d3	Mar.w4, d5	Prachi, Isha	2 person per week
Testing	Mar.w4, d5	Apr.w1, d1	Apr.w1.d4	Apr.w1.d5	Isha, Prachi	2 person per week
References	Mar.w2,d3	Mar.w3, d1	Mar.w3, d3	Mar.w3, d5	Akriti	1 person per week
Annexure	Mar.w2,d3	Mar.w3, d1	Mar.w3, d3	Mar.w3, d5	Akriti	1 person per week

Table 5.1 Project Scheduling

5.2 TIMELINE CHART

WORK TASKS	JANUARY				FEBRUARY				MARCH				APRIL			
	W 1	W 2	W 3	W 4	W 1	W 2	W 3	W 4	W 1	W 2	W 3	W 4	W 1	W 2	W 3	W 4
Problem Statement																
Software Lifecycle Model																
Project Scheduling																
Timeline Chart																
SRS																
Context Level Diagram																
DFD 1																
DFD 2																

Data Dictionary																	
Use Case Diagram																	
Use Case Description																	
Project Metrics																	
Effort estimation (COCOMO)																	
Risk Analysis																	
Data Design																	
Component Design																	
Testing																	

Table 5.2 Timeline Chart

5.3 EFFORT ESTIMATION & FP-BASED COMPUTING

- Function Point Metric is an example of Product metrics for Analysis Model.
- It is used as a means for measuring the functionality delivered by a system and also examines the requirement/ analysis model for predicting size of resultant system.
- Using historical data, Function Point metric can be used to:-

- i. Estimate the effort or cost required to design, code or test the software.
 - ii. To predict the number of errors that will be encountered during testing.
 - iii. Forecast number of components or number of projected source links in the implemented system. Function points are derived using empirical relationships based on countable (direct) measures of software's information domain and quantitative assessment of software complexity.
- Software Information Domain Values consists of number of: -

- i. External Inputs (EI)
- ii. External Outputs (EO)
- iii. External Query (EQ)
- iv. Internal Logical Files (ILF)
- v. External Interface Files (EIF)

To compute function points (FP), the following relationship is used:

$$\text{FP} = \text{Count total} \times [0.65 + 0.05 \times \sum (Fi)],$$

where Count total= Sum of all Function Point entries

Calculation of Value Adjustment Factors (VAF) is based on the responses of the following questions:

1	Does the system require reliable backup and recovery?	3
2	Are specialized data communications required to transfer the information to and from the application?	2
3	Are there distributed processing functions?	3
4	Is performance critical?	3
5	Will the system work in an existing heavily utilized operational environment?	3

6	Does the system require online data entry?	5
7	Does the online data entry require input transactions to be built over multiple screens or operations?	5
8	Are Internal Logical Files updated online?	5
9	Are input-output queries or files complex?	2
10	Is the internal processing complex?	4
11	Is the code designed to be reusable?	4
12	Are conversion and installation included in design?	1
13	Is the system designed for multiple installations in multiple organizations?	3
14	Is the application designed to facilitate changes and ease of use by the user?	4
COUNT TOTAL (ΣF_i)		47

Table 5.3 Value Adjustment Factor (VAF)

The count total is the sum of all FP entries obtained from the following table:

INFORMATION DOMAIN VALUES	COUNT	WEIGHING FACTOR			COUNT* WEIGHING FACTOR (SIMPLE)
			AVERAGE	COMPLEX	
External Inputs (EI)	25	3	4	6	75
External Outputs (EO)	12	4	5	7	48
External Queries (EQ)	1	3	4	6	3
Internal Logical Files (ILF)	10	7	10	15	70

External Interface Files (EIF)	1	5	7	10	5
TOTAL COUNT					201

Table 5.4: Weighting factor of information domain values

$$\begin{aligned}
 \text{FUNCTION POINT (FP)} &= \text{Total Count} \times [0.65 + (0.01 \times \Sigma (Fi))] \\
 &= 201 \times [0.65 + (0.01 \times 49)] \\
 &= 229.14
 \end{aligned}$$

5.4 COST ESTIMATION: COCOMO II MODEL

Barry Boehm gave a hierarchy of software estimation models called COCOMO i.e. constructive cost model. The original COCOMO model was widely used in the industry and was later evolved into a comprehensive model. The estimation model is called COCOMO II Model.

COCOMO II is a hierarchy of estimation models that consists of:

- 1. Application Composition Model:** It is used during the prototyping of user interfaces, assessment of process during system and software interaction and evaluation of technology maturity.
- 2. Early Design Stage Model:** It is used once. The requirement has been stabilized and basic software architecture is established.
- 3. Post-Architecture Stage Model:** This model is used during the construction of software.

• Application Composition Model

These models use sizing information for which 3 options are available which are object points, function points and lines of source code.

Object-Point: Object Point is an indirect software measure computed using counts of number of screens on the user interface, number of reports generated and number of reusable components and 3GL components required to build the application. Each object instance is classified into one of the three complexities.

levels: Simple, Medium or Difficult. Complexity is a function of number and source of client and server data tables required to generate a screen or report and number of views or sections presented as part of the screen or report.

Complexity weighting for object types:

OBJECT-TYPE	COMPLEXITY WEIGHT		
	Simple	Medium	Difficult
Screens	1	2	3
Reports	2	5	8
3GL Components			10

Table 5.5 Complexity Weight for Object Type

Productivity rate for object points:

Developer's experience or capability	Very Low	Low	Normal	High	Very High
Environment maturity or capability	Very Low	Low	Normal	High	Very High
Productivity	4	7	13	25	50

Table 5.6 Productivity Weight for Object Point

$$\begin{aligned}
 \text{Object Point Count} &= \text{an original number of object instances} * \text{Weighting Factor} \\
 (\text{Simple}) &= (17*2) + (0*5) \\
 &= 34
 \end{aligned}$$

0% of the components are reusable.

NOP = New Object Points (or Object Point Counts)

$$= (\text{Object Points}) * [(100 - \% \text{ of reuse}) / 100]$$

$$= 34 * [(100 - 0) / 100]$$

$$= 34$$

The developer's experience and capability in a similar environment is low.

PROD (Productive Rate) = 7

$$\text{Estimated Effort} = \text{NOP} / \text{PROD}$$

$$= 2.4285714 = \mathbf{2.43 \text{ PM}}$$

The total number of screens will be more in number while actually building the application. Here, only some of the sample screens are shown, and our effort calculated is 2.43 PM which is according to screens only.

5.5 RISK ANALYSIS

SOFTWARE RISKS

It involves 2 aspects: uncertainty and loss. Uncertainty means risk may or may not happen. If the risk becomes reality, then unwanted consequences or loss will occur.

PHASES INVOLVED IN RISK ANALYSIS AND MANAGEMENT

Risk Identification

Risk Analysis

Risk ranking and assessment

Creating risk plan or RMMM plan

TYPES OF RISKS

According to general categorization there are 3 types of risks:

Known Risk

Predictable Risk

Unpredictable Risk

Another category of risk type:

Project Risk

Technical Risk

Business Risk

Project Risk: Identify potential problems that might occur in budget, schedule and staffing. It also includes project complexity, project size and degree of structural uncertainty.

Technical Risk: Identify potential design problem, implementation problem, interface problems, verification problem and maintenance problem. They threaten the quality of the software produced.

Business Risk: Threatens the viability of the software to be built and often jeopardizes the project or the product. There are 5 types of business risks:

Market Risk

Strategic Risk

Sales Risk

Management Risk

Budget Risk

ASSESSING OVERALL PROJECT RISK

1. Have top software and customer managers formally committed to supporting the project? **YES,**
2. Are end-users enthusiastically committed to the project and the system/product to be built? **YES,**
3. Are requirements fully understood by the software engineering team and its

customer? **YES,**

4. Have customers been involved fully in the definition of requirements? **YES,**
5. Do end-users have realistic expectations? **YES,**
6. Is the project scope stable? **YES,**
7. Does the software engineering team have the right mix of skills? **YES,**
8. Are project requirements stable? **YES,**
9. Does the project team have experience with the technology to be implemented?
YES,
10. Is the number of people on the project team adequate to do the job? **YES,**
11. Do all customer/user constituencies agree on the importance of the project and
on the requirements for the system/product to be built? **YES,**

6. DESIGN

6.1 DATA DESIGN

CUSTOMER / ADMIN

FIELD NAME	TYPE	SPECIFICATION	CONSTRAINTS	UNIQUE	DESCRIPTION
Name	Character	10 alphabetic characters	Not null	No	Customer's name
Phone No	Integer	10 Integer characters	Not null	Yes	Login phone no. of the user must contain 10-digit numbers.
Password	Character	10 alphanumeric characters	Not null	Yes	Login password of the user must contain 8 characters including a special one.
Email Id	Varchar	20 alphanumeric characters	Not null, primary key	Yes	Gives email Id of the user.

Table 6.1 DATA DESIGN : CUSTOMER / ADMIN

FEEDBACK

FIELD NAME	TYPE	SPECIFICATION	CONSTRAINTS	UNIQUE	DESCRIPTION
Rating	Integer	1 integer character	No	NO	You can provide the app the rating by the scale of 5
Comments	String	50 alphanumeric characters	No	NO	User can talk about his/her experience.

Table 6.1 DATA DESIGN : FEEDBACK

BILL

Field Name	Type	Specification	Constraint	Unique	Description
Mode of Payment	Character	10 alphabetic characters	Not Null,	Yes	Gives the mode of payment which the user prefers
Card Number	Integer	12 integer character	Not Null, Primary Key	Yes	Card Number of the customer
Expiry Date	Date	MM/DD/YY Format	Not Null	Yes	Expiry Date of the Customer's card
CVV	Integer	3 integer characters	Not Null	Yes	Gives the CVV code of the customer's card
Date	Date	MM/DD/YY Format	Not Null	No	The date at which the payment is being done

Table 6.1 DATA DESIGN : BILL

7. TESTING

We are performing **White Box Testing** to **Book a Room**.

PSEUDOCODE FOR ROOM BOOKING

1. THE ROOM BOOKING PROCEDURE BEGINS

2. Enter choice
3. If (choice==" Book a room")

THEN

4. Enter the number of rooms
Enter room type
Enter check-in date
Enter check-out date
5. If (room type==" Single/Couple" & number of rooms == available rooms)

THEN

6. Generate booking request
7. If (REQUEST ACCEPTED)
8. Store booking details in the booking database Allocate number of rooms according to room type in room database
9. Else
10. Show denies request
11. End if
12. Else
13. Entered wrong details
14. End if
16. Else
17. If (previous booking present)

18. Print booking details
19. Else
20. No data was found
21. End if
22. End if
23. END PROCEDURE

FLOWGRAPH

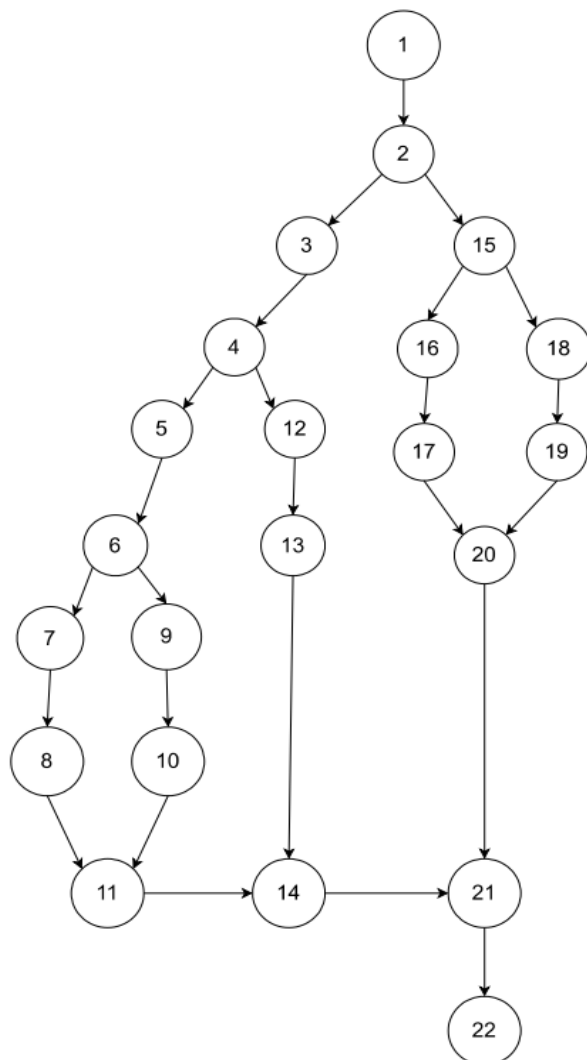


Figure 7.1 FLOWGRAPH

CYCLOMATIC COMPLEXITY OF RESULTANT GRAPH

$V(G)$ = Number of regions

$$= 5$$

$V(G)$ = Edges - Nodes + 2

$$= 25 - 22 + 2$$

$$= 3 + 2$$

$$= 5$$

$V(G)$ = Predicate nodes + 1 (Predicate nodes = 2, 4, 6, 15)

$$= 4 + 1$$

$$= 5$$

INDEPENDENT PATHS

Path 1: 1-2-3-4-5-6-7-8-11-14-21

Path 2: 1-2-3-4-5-6-9-10-11-14-21

Path 3: 1-2-3-4-12-13-14-21

Path 4: 1-2-15-16-17-20-21

Path 5: 1-2-15-18-19-20-21

Test ID	Input Values	Actual Output	Expected Output
1	Rooms are booked (request accepted)	To be observed after execution	Display the booked room
2.	Booking is not booked	To be observed after execution	Display booking cannot be done
3.	Room are not available	To be observed after execution	Display rooms are not available
4.	Booking History	To be observed	Show previous

	is present	after execution	booking history
5.	Booking History is not present	To be observed after execution	Show no previous booking details present

Table 7.1 TEST CASE

CODING OF BOOK A ROOM MODULE

(i) bookRoom.php

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3"
crossorigin="anonymous">
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IIRH9sENBO0LRn5q+8nbTov4+1p"
crossorigin="anonymous"></script>
<style>
  body{
    background-color: rgb(115, 115, 116);
  }
  div{
```

```

    width: 20%;
    margin: auto;
    margin-top: 200px;
    justify-content: center;
}

.btnn{
    border-radius: 10px;
    padding: 20px 130px;
    margin-bottom: 20px;
    color: rgb(2, 28, 51);
    font-weight: 900;
}

.btnn2{
    border-radius: 10px;
    padding: 20px 80px;
    color: rgb(1, 19, 34);
    font-weight: 900;
}

h1{
    color: aliceblue;
}
</style>
</head>
<body>
    <div>
        <h1>Welcome to Dusk Till Dawn Hotel !!</h1>
        <button type="button" class="btn btn-warning btnn">Book New Room</button>
        <button type="button" class="btn btn-info btnn2">View Booking History</button>

    </div>
</body>

```



```
</html>
```

(ii) newRoom.php

```
<?php include 'config.php';?>
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Book New Room</title>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
1BmE4kWBq78iYhFIdvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3"
crossorigin="anonymous">
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
ka7Sk0Gl4gmtz2MlQnikT1wXgYsOg+OMhuP+IIRH9sENBO0LRn5q+8nbTov4+1p"
crossorigin="anonymous"></script>
<style>
body{
  margin: 30px;
}
form{
  padding: 20px;
  width: 60%;
  margin: auto;
  padding-bottom: 40px
}
```

```

#cars{
  padding: 10px;
  width: 100%;
}

.container{
  box-sizing: border-box;
  border: 10px solid rgb(112, 112, 112);
  background-color: rgb(183, 0, 255);

}

h1{
  color: white;
  font-size: 86px;
}

button{
  width: 100%;
  margin-top: 10px;
}

label{
  font-weight: 900;
  color: white;
}

</style>
</head>
<body>
<div class="container">
  <form>
    <h1>Book A Room</h1>
    <div class="mb-3">
      <label for="exampleInputEmail1" class="form-label">Room Type</label>
      <div class="mb-3">

```

```

<select id="cars" name="cars" name="roomtype">
  <option value="volvo">Select Your Room Type </option>
  <option value="volvo">Single</option>
  <option value="saab">Couple</option>
  <option value="fiat">Triple</option>
  <option value="audi">Quad</option>
  <option value="volvo">Studio</option>
  <option value="volvo">Queen</option>
  <option value="saab">King</option>
  <option value="fiat">Twin</option>
  <option value="audi">Double-double</option>
</select>
</div>
</div>

<div class="mb-3">
  <label for="exampleInputEmail1" class="form-label">Number of Rooms</label>
  <input type="text" name="roomNo" class="form-control" id="exampleInputEmail1"
aria-describedby="emailHelp">

</div>

<div class="mb-3">
  <label for="exampleInputEmail1" class="form-label">Check In</label>
  <input type="date" name="checkIn" class="form-control" id="exampleInputEmail1"
aria-describedby="emailHelp">

</div>

<div class="mb-3">
  <label for="exampleInputEmail1" class="form-label">Check Out</label>

```

```

        <input type="date" name="checkOut" class="form-control" id="exampleInputEmail1"
aria-describedby="emailHelp">

    </div>

    <button type="submit" class="btn btn-primary" name="submit">Submit</button>
</form>
</div>
</body>
</html>

```

(iii) previousRoom.php

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <style>
        body{
            background-color: cornflowerblue;
            margin: 50px;
        }
        table{
            border: 4px solid rgb(26, 26, 26);
            border-collapse: collapse;
        }
    </style>

```

```

    td{

        border: 5px solid rgb(37, 37, 37);
        padding: 10px;

    }
</style>
</head>
<body>

    <?php

$server='localhost';
$username='root';
$password="";
$database='users';

$conn=mysqli_connect($server, $username, $password, $database);
if ($conn->connect_error) {
    die("connection failed : ".$conn->connect_error);
}
echo"";

    $sql="SELECT * FROM bookRoom";
    $result=mysqli_query($conn,$sql);
    $i=0;
    if ($result->num_rows>0) {
        while ($rows=$result->fetch_assoc()) {
            echo'

            <tr>

                <th scope="row">'.++$i.'</th>

                <td>'.$rows['No. of Room'].'</td>

                <td>'.$rows['Room Type'].'</td>

                <td>'.$rows['Location'].'</td>

```

```

        <td>'.$rows['Check In'].'</td>
        <td>'.$rows['Check Out'].'</td>
    </tr>
    ';
    }
}
else {
    echo "";
}
?>

</tbody>
</table>
</div>

</body>
</html>

```

(iv) config.php

```

<?php
$server='localhost';
$username='root';
$password="";
$databse='user';

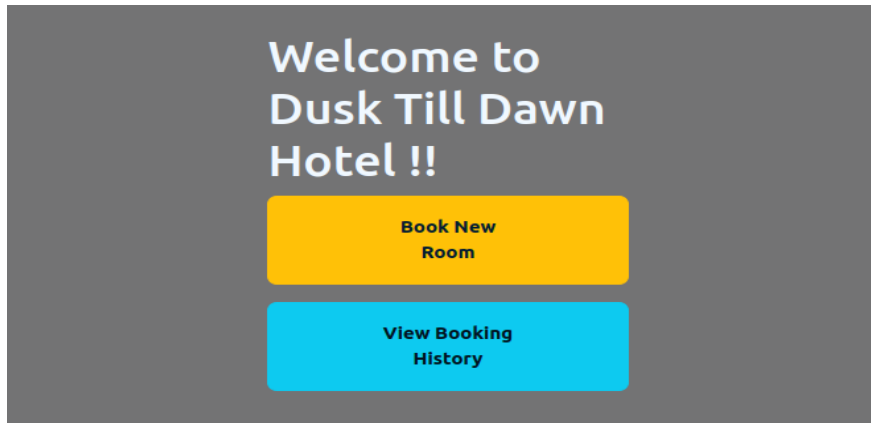
$conn=mysqli_connect($server, $username, $password, $databse);

if ($conn->connect_error) {
    die("connection failed : ".$conn->connect_error);
}
echo "";

```

```
if(isset($_POST['submit'])) {  
    $roomtype=$_POST['roomtype'];  
    $roomNo=$_POST['roomNo'];  
    $checkIn=$_POST['checkIn'];  
    $checkOut=$_POST['checkOut'];  
  
    $sql="INSERT INTO `user`(`ROOM TYPE`,`NO. OF ROOM`,`CHECK IN`,`CHECK  
OUT`) VALUES ('$roomNo','$checkIn','$checkOut')";  
    if (mysqli_query($conn,$sql)) {  
        echo "";  
    }  
    else {  
        echo "ERROR : Not Able to execute $sql. ".mysqli_error($conn);  
    }  
}  
  
mysqli_close($conn);  
  
?>
```

USER INTERFACE



7.2 Figure Display UI

Booking History

No. of Rooms	Room Type	Check In	Check Out
1	Double	02/02/22	02/02/22

7.3 Figure Booking History UI

Book A Room

Room Type

Select Your Room Type ▼

Number of Rooms

Check In

dd/mm/yyyy

Check Out

dd/mm/yyyy

Submit

7.4 Figure Book A Room UI

8. ANNEXURES (SAMPLE SCREENS)



Figure 8.1 Display Screen

A registration form overlay on a background image of a restaurant interior. The form has a title "Register" in a cursive font. It contains three input fields: "Email", "Phone No.", and "Password", each with a corresponding icon (envelope, phone, and key respectively). Below the fields is a yellow "Continue" button.

Figure 8.2 Registration-1 Screen

A screen showing a message "One Time Password is sent to your registered phone number." in a cursive font. Below the message is a form with four input boxes for the OTP and a "Resend OTP" button.

Figure 8.3 Registration-2 Screen

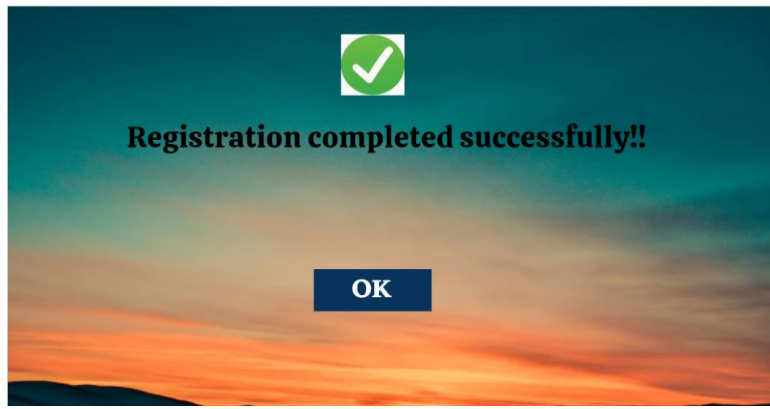


Figure 8.4 Registration-3 Screen

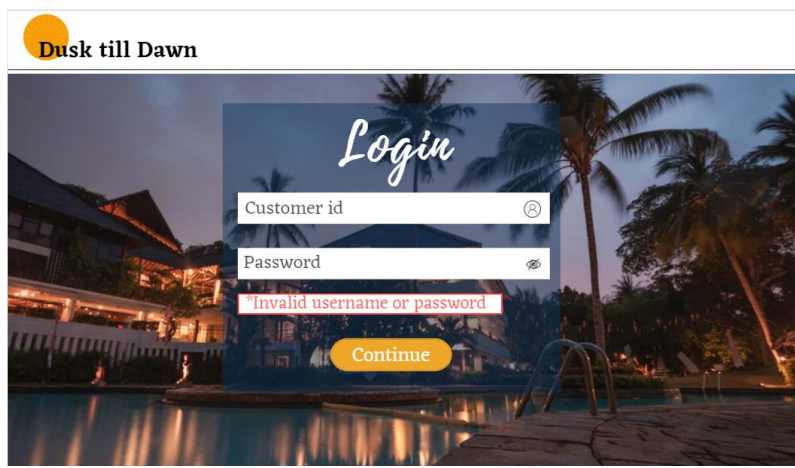


Figure 8.5 Login Screen

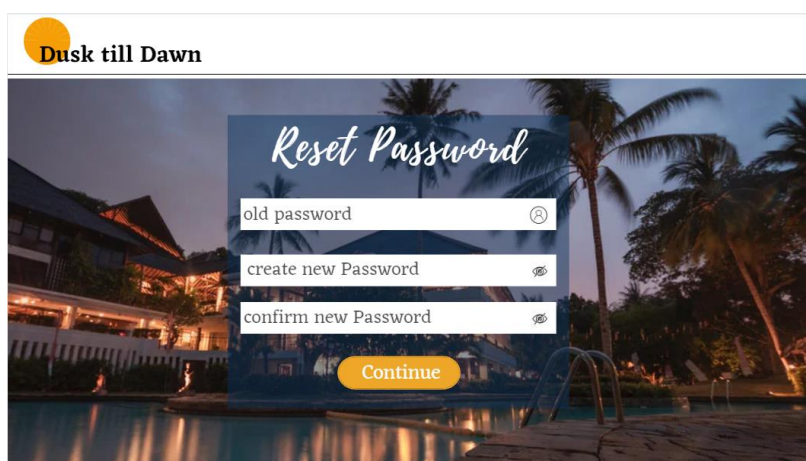


Figure 8.6 Reset Password-1 Screen

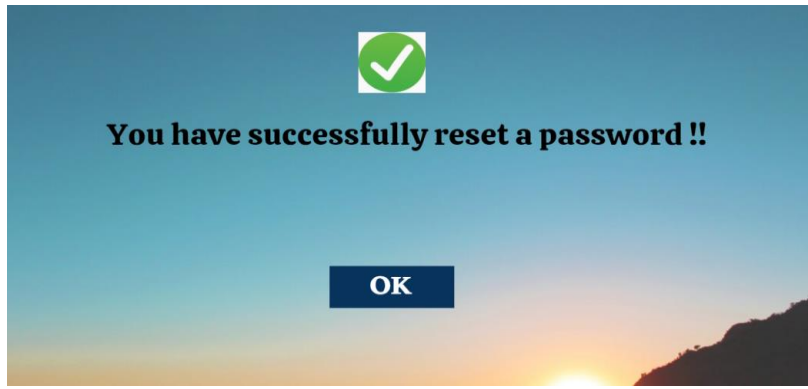


Figure 8.7 Reset Password-2 Screen

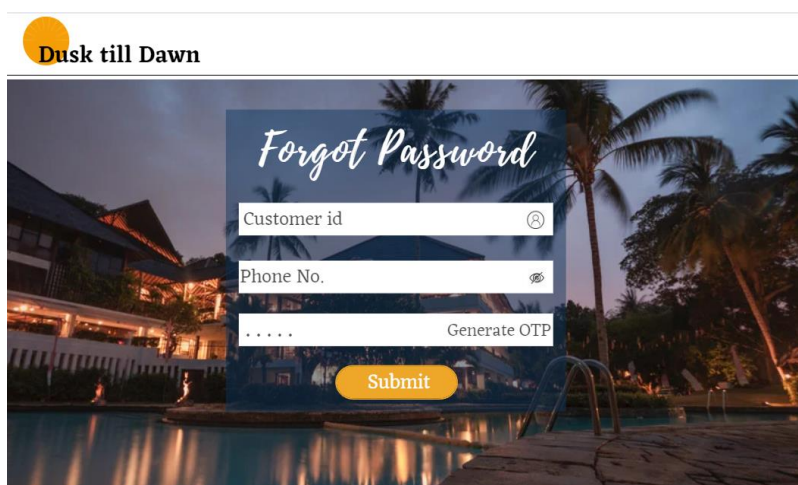


Figure 8.8 Forgot Password Screen

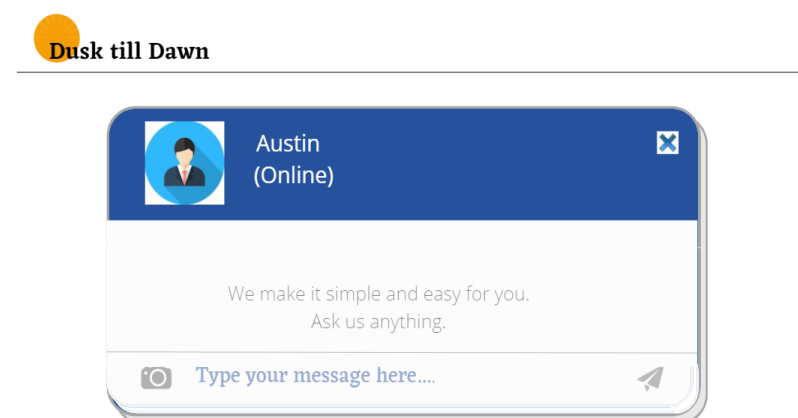


Figure 8.9 Chatbox Screen

Dusk till Dawn

Receipt

Deluxe Room

Number of Rooms	1
Room Number	454
Customer id	23456
Check in	Mon 25/04/22
Check out	Thu 28/04/22
Total guests	2
Total charges	INR.3000
Grand Total	INR.3500
Online payment	INR.3600

Cancellation of Rooms

Customer Id

Cancellation Period

XX Days

Refundable Amount

INR.XXXX

Cancel

Confirm

Figure 8.10 Booking Cancellation-1 Screen

Dusk till Dawn

Your cancellation request has been accepted.

You will be notified soon...

Figure 8.11 Booking Cancellation-2 Screen

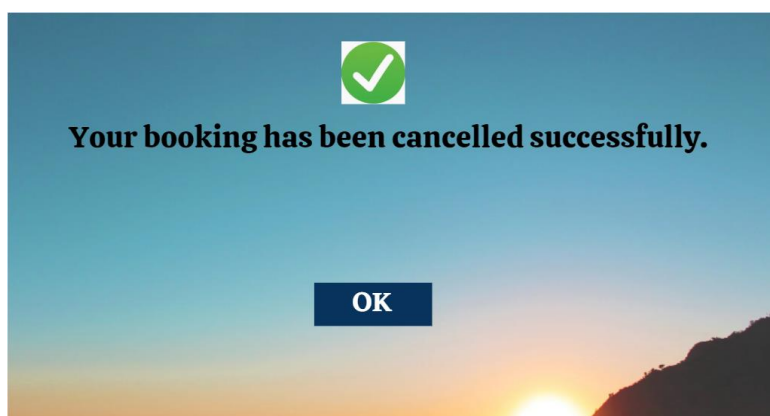


Figure 8.12 Booking Cancellation-3 Screen


Modify Room Details

Number of Rooms


Room Type

Total Guest

Available Rooms



Deluxe Room
A beautifully design cozy rooms 137 sq m, they have a fully equipped bathroom with separate bathtub and modern rain shower, a spacious work area with triple play technology on fibre, integrated IP technology, in room minibar and high 40-inch LED screen



Quad Room
A Quad rooms with 2 separate bedrooms sized at approximately 25 m 2/270sq ft. size of a quadruple room with all beds in one room is approximately 27 m2/290 sq ft. This size is approximate and as it is a victorian style building, the sizes of room vary.

Figure 8.13 Booking Modification Screen

Receipt

Deluxe Room

Number of Rooms	2
Room Number	454
Customer id	Cid23456
Check in	Mon 25/04/22
Check out	Thu 28/04/22
Total guests	4
Total charges	INR.3000
Grand Total	INR.3500
Online payment	INR.3600

Payment





Credit/Debit Card	Mobile Banking	Internet Banking
Cardholder Name <input type="text"/>	Card Name <input type="text"/>	
Expiration Month <input type="text"/>	Year <input type="text"/>	CVV <input type="text"/>
		
		
Pay		

Figure 8.14 Payment-1 Screen

Your transaction is processing.....




Figure 8.15 Payment-2 Screen



Figure 8.16 Payment-3 Screen

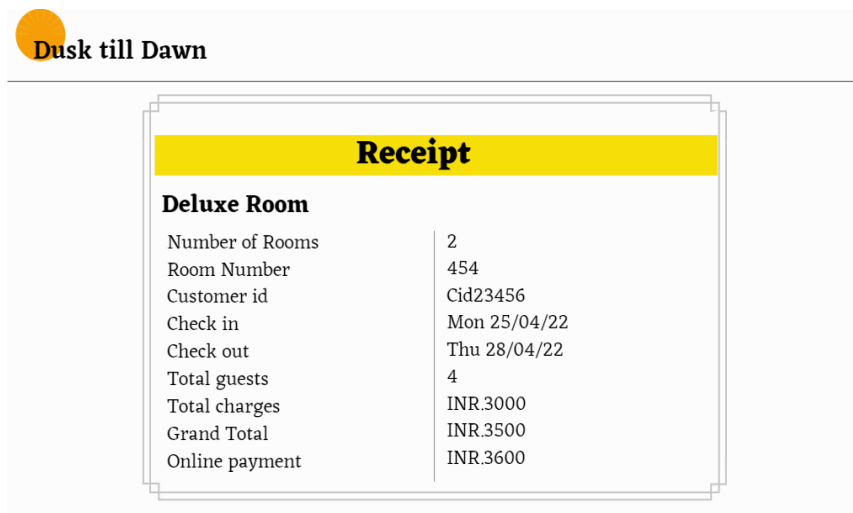


Figure 8.17 Payment-4 Screen

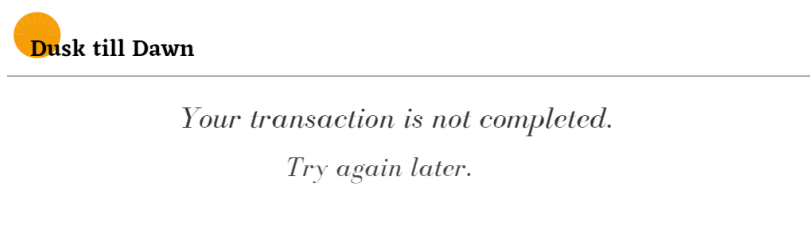
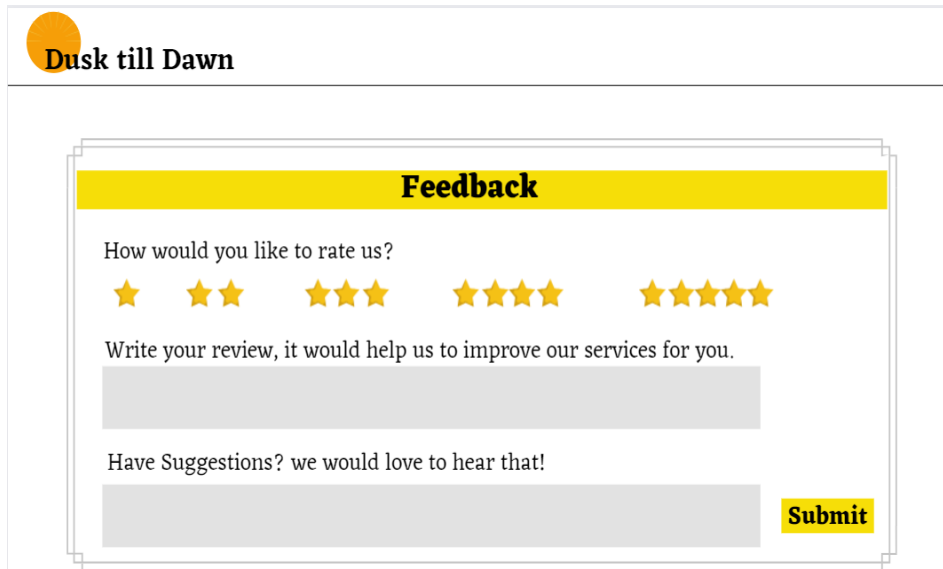


Figure 8.18 Payment-5 Screen



Dusk till Dawn

Feedback

How would you like to rate us?

★ ★★ ★★★★★

Write your review, it would help us to improve our services for you.

Have Suggestions? we would love to hear that!

Submit

Figure 8.19 Feedback Screen

8. REFERENCES

Software Engineering, Ian Sommerville

Requirements Engineering: <https://morse.inf.unideb.hu/valseg/gybitt/07/ch02.html>

Room Booking System: <https://www.scribd.com/doc/63824633/Room-Booking-System>

Case Study: <https://www.scribd.com/doc/27927992/Room-Booking-Case-Study>

Requirement Engineering: https://en.wikipedia.org/wiki/Requirements_engineering

R.S. Pressman, Software Engineering: A Practitioner's Approach, McGraw-Hill

P. Jalote, An Integrated Approach to Software Engineering, Narosa Publishing House

<https://www.smartics.eu/confluence/display/PDACL/How+to+document+a+Software+Development+Project>