High Level Design & Low Level Design

The purpose of this document is to provide with a template for documenting both HLD & LLD.

**Document Control :**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Revision History** | | | | | | | | |
|  |  |  | |  |  |  |  |  |
| **Date** | **Version** | **Author** | **Brief Description of Changes** | | | | **Approver Signature** | |
|  | 1.0 | GROUP5 |  | | | |  | |
|  |  |  |  | | | |  | |
|  |  |  |  | | | |  | |

[1. Introduction 3](#_Toc368912248)

[1.1. Intended Audience 5](#_Toc368912249)

[1.2. Acronyms/Abbreviations 5](#_Toc368912250)

[1.3. Project Purpose 5](#_Toc368912251)

[1.4. Key Project Objectives 5](#_Toc368912252)

[1.5. Project Scope and Limitation 6](#_Toc368912253)

[1.5.1. In Scope 6](#_Toc368912254)

[1.5.2. Out of scope 6](#_Toc368912255)

[1.6. Functional Overview 6](#_Toc368912256)

[1.7. Assumptions, Dependencies & Constraints 7](#_Toc368912257)

[1.8. Risks 7](#_Toc368912258)

[2. Design Overview 7](#_Toc368912259)

[2.1. Design Objectives 7](#_Toc368912260)

[2.2. Architectural Strategies 7](#_Toc368912262)

[2.2.1. Design Alternative 8](#_Toc368912263)

[2.2.2. Reuse of Existing Common Services/Utilities 8](#_Toc368912264)

[2.2.3. Creation of New Common Services/Utilities 8](#_Toc368912265)

[2.2.4. User Interface Paradigms 8](#_Toc368912266)

[2.2.5. System Interface Paradigms 8](#_Toc368912267)

[2.2.6. Error Detection / Exceptional Handling 9](#_Toc368912268)

[2.2.7. Memory Management 9](#_Toc368912269)

[2.2.8. Performance 9](#_Toc368912270)

[2.2.9. Security 10](#_Toc368912271)

[2.2.10. Concurrency and Synchronization 10](#_Toc368912272)

[2.2.11. Housekeeping and Maintenance 10](#_Toc368912273)

[3. System Architecture 10](#_Toc368912274)

[3.1. System Architecture Diagram. (Not Necessary) 10](#_Toc368912275)

[3.2. System Use-Cases 10](#_Toc368912276)

[3.3. Subsystem Architecture 13](#_Toc368912277)

[3.4. System Interfaces 17](#_Toc368912278)

[3.4.1. Internal Interfaces 17](#_Toc368912279)

[3.4.2. External Interfaces 17](#_Toc368912280)

[4. Detailed System Design 18](#_Toc368912281)

[4.1. Key Entities 18](#_Toc368912282)

[4.2. Detailed-Level Database Design 18](#_Toc368912283)

[4.2.1. Data Mapping Information 18](#_Toc368912284)

[4.2.2. Data Conversion 18](#_Toc368912285)

[4.3. Archival and retention requirements 18](#_Toc368912286)

[4.4. Disaster and Failure Recovery 18](#_Toc368912287)

[4.5. Business Process workflow 19](#_Toc368912288)

[4.6. Business Process Modeling and Management (as applicable) 19](#_Toc368912289)

[4.7. Business Logic 19](#_Toc368912290)

[4.8. Variables 22](#_Toc368912291)

[4.9. Activity / Class Diagrams (as applicable) 23](#_Toc368912292)

[4.10. Data Migration 24](#_Toc368912293)

[4.10.1. Architectural Representation 24](#_Toc368912294)

[4.10.2. Architectural Goals and Constraints 24](#_Toc368912295)

[4.10.3. Logical View 24](#_Toc368912296)

[4.10.4. Architecturally Significant Design Packages 24](#_Toc368912297)

[4.10.5. Data model 24](#_Toc368912298)

[4.10.6. Deployment View 24](#_Toc368912299)

[5. Environment Description 3](#_Toc368912300)

[5.1. Time Zone Support 25](#_Toc368912301)

[5.2. Language Support 25](#_Toc368912302)

[5.3. User Desktop Requirements 25](#_Toc368912303)

[5.4. Server-Side Requirements 25](#_Toc368912304)

[5.4.1. Deployment Considerations 25](#_Toc368912305)

[5.4.2. Application Server Disk Space 25](#_Toc368912306)

[5.4.3. Database Server Disk Space 25](#_Toc368912307)

[5.4.4. Integration Requirements 26](#_Toc368912308)

[5.4.5. Jobs 26](#_Toc368912309)

[5.4.6. Network 26](#_Toc368912310)

[5.4.7. Others 26](#_Toc368912311)

[5.5. Configuration 26](#_Toc368912312)

[5.5.1. Operating System 26](#_Toc368912313)

[5.5.2. Database 26](#_Toc368912314)

[5.5.3. Network 26](#_Toc368912315)

[5.5.4. Desktop 26](#_Toc368912316)

[6. References 26](#_Toc368912317)

[7. Appendix 26](#_Toc368912318)

# 

# Introduction

The aim of the project “Hotel Reservation/ management software” is to provide an online for reservation. This application provides the availability of several types of rooms, prices, complimentary, number of the available rooms and hotel’s address. The system will be so simple and attractive which will make the customer comfortable to use and choose their ideal room. The final bill will be shown to the client.

## Intended Audience

|  |  |
| --- | --- |
| Read by | Client |
| Used by | Maintenance team, Development Team, Clients |
| Accessed by | General purpose application, anyone can access it |

## Acronyms/Abbreviations

|  |  |
| --- | --- |
| UT | Unit Test |
| IT | Integrated Test |
| DFD | Data Flow Diagram |
| ERD | Entity Relationship Diagram |

## Project Purpose

The purpose of this project is to increase and make easier of reserving rooms of a hotel to customer. The customer can reserve a room in a hotel without any difficulties, paperwork and any involvement of the hotel management. Everything can be done remotely with ease.

## Key Project Objectives

* A hotel reservation system enables clients to schedule their length of stay, room selection and payment all in one place.
* Overall system quality should be good and it should be a smooth experience for both customers and administrators.
* The system should be easy to maintain and navigate. Object oriented programming techniques are used in the project.
* The system will remain up at all times. The hardware and software should be robust .

## Project Scope and Limitation

The Hotel Management System is to simplify the day-to-day processes of the hotel. The system will be able to handle many services to take care of all customers in a quick manner by providing multiple client facility by the server. The client-side operation will display a menu-based interface to select their requirement.

The project will not show the high-end usage of the real-time reservation but a part of simpler prototype. It does not give a detailed information of customers and the scope for modifying details.

### In Scope

Clients are made convenient to book a room if there is a need instead of calling up to the hotel to enquire details and struggle to find a suitable slot to book. A user-friendly menu-based interface is provided to make it convenient for customers/clients to use the application. The intended room type, number of days of stay are provided by the client followed by which the management/ server provides them with suitable data to select from.

The Hotel Management System is to simplify the day-to-day processes of the hotel. The system will be able to handle many services to take care of all customers in a quick manner by providing multiple client facility by the server.

### Out of scope

The real-time replica of hotel reservation system cannot be provided as this just a part of a simpler prototype of the real-time use.

## Functional Overview

The menu selection feature is provided for the client. Single client or multiple clients can login into the system and select their required option. The server side validates the client and processes the information sent by the clients. This processing can be done by defining the type of message being sent and received by both client and server using TLV. The server sends responses for respective received messages. The server also provides error messages in few instances. The whole project will be built on TCP client-server model.

## Assumptions, Dependencies & Constraints

* Should be UNIX based/ should have any linux/putty installed.
* Assume that source code occupies more than 250kb and less than 600kb

**Constraints:** Not applicable for this project

## Risks

The potential risks are changes in market, customer satisfaction.

# Design Overview

This section gives a brief description of flow of data and application. It gives an idea of features required by the application and can be described as visual aid of the project flow. It is a road-map of a contingency plan of the project.

## Design Objectives

* Provide the developer with a list of requirements.
* Give an idea on various features required by the application.
* Describe the flow of work.
* Define the data flow.

### Recommended Architecture

The recommended Architecture is the level-0 Data Flow Diagram, level-1 Data Flow Diagram and Flow Diagram. These give a necessary and detailed information on requirements, necessities, practically involved applications etc.

## Architectural Strategies

DFD-0, DFD-1, ER diagram, flow diagram and class diagram can be considered as main architectural strategies.

* **DFD-0:** It gives information of the top-level flow of an application. In this case, it describes general flow of work done in the Hotel management system
* **DFD-1:** It gives information on the next level flow of the application. In this case, it gives information on flow of work in hotel management, how can a client connect to hotel server and reserve a room.
* **ER Diagram:** It shows various entities involved in a hotel management system. It also shows the entity relationship between each entity, by which the developer will have a detailed understanding of the application/project.
* **Flow Diagram:** It shows the flow of work.
* **Class Diagram:** It gives an overall description of various classes involved in the development phase of the project.

### Design Alternative

Not applicable for this project.

### Reuse of Existing Common Services/Utilities

For design and development of the project the existing common services are used.

**Design Phase:** Star UML

**Development:** kali linux/putty

### Creation of New Common Services/Utilities

Not applicable for this project

### User Interface Paradigms

* Client
* Server

### System Interface Paradigms

**System Features**

**G5\_FR01: Online System for Booking Rooms**

It is online hotel reservation system where users can access and check for availability rooms.

**G5\_FR02: Files**

The server maintains all the data related to a hotel booking system i.e., rooms/categories/price etc.

**G5\_FR03: Booking Room**

The server provides details as requested by the client and proceeds with booking if confirmed.

**G5\_FR04: Supporting System for Multiple Client Login**

Multiple clients can connect to the server and try to reserve the same/different category of rooms or services. Server uses appropriate protection mechanism for providing this support and avoid corruption.

**G5\_FR05: Availability of Rooms or Services**

The server provides statistics related to availability of rooms/services etc...

**G5\_FR07: Authentication**

The client needs to be authorized by the server before accessing the hotel database.

**G5\_FR10: Confirm/Cancel Booking**

The clients have the option to confirm the reservation or cancel the reservation.

### Error Detection / Exceptional Handling

Proper error detection is maintained throughout the code. Error/ Invalid messages are displayed wherever necessary. Debugging is done wherever necessary to handle the errors/ exceptions. File handling is used to authenticate the client login in which the exception handling is used.

### Memory Management

The application involves stack memory allocation. In stack memory allocation, the size of memory to be allocated is known to the compiler and whenever a function is called, its variables get memory allocated on the stack. And whenever the function call is over, the memory for the variables is de-allocated. It is best way to handle memory management and is being implemented in the current project.

### Performance

The application/ system is fast and responsive. It is a full-fledged application with limited functionality.

### Security

Security cannot be guaranteed in this project because of it’s simple application and simple source code.

### Concurrency and Synchronization

Concurrency and synchronization are the main segments of any code. The application can have multiple clients accessing it, if there is no concurrency used problems arise. If the message intended to communicate between server and client has any problems, then synchronization must be used.

### Housekeeping and Maintenance

Makefile and valgrind can be used for cleanup and memory leak checks.

The rest is not applicable to this project.

# System Architecture

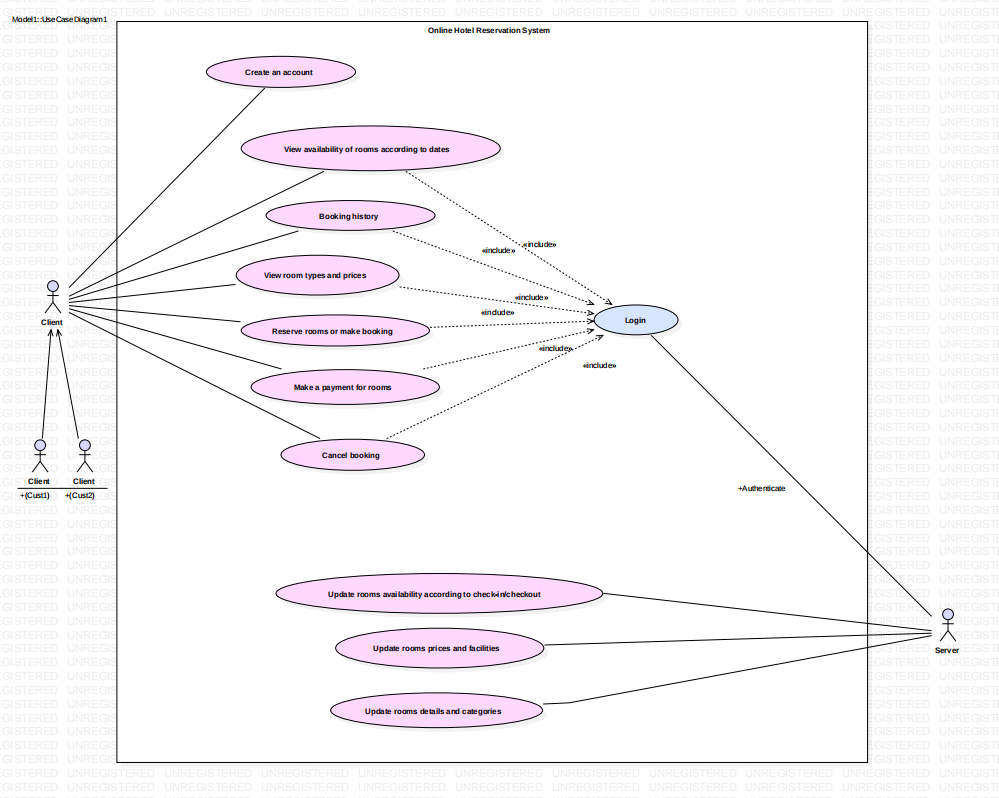
## System Architecture Diagram. (Not Necessary)

Not applicable to this project.

## System Use-Cases

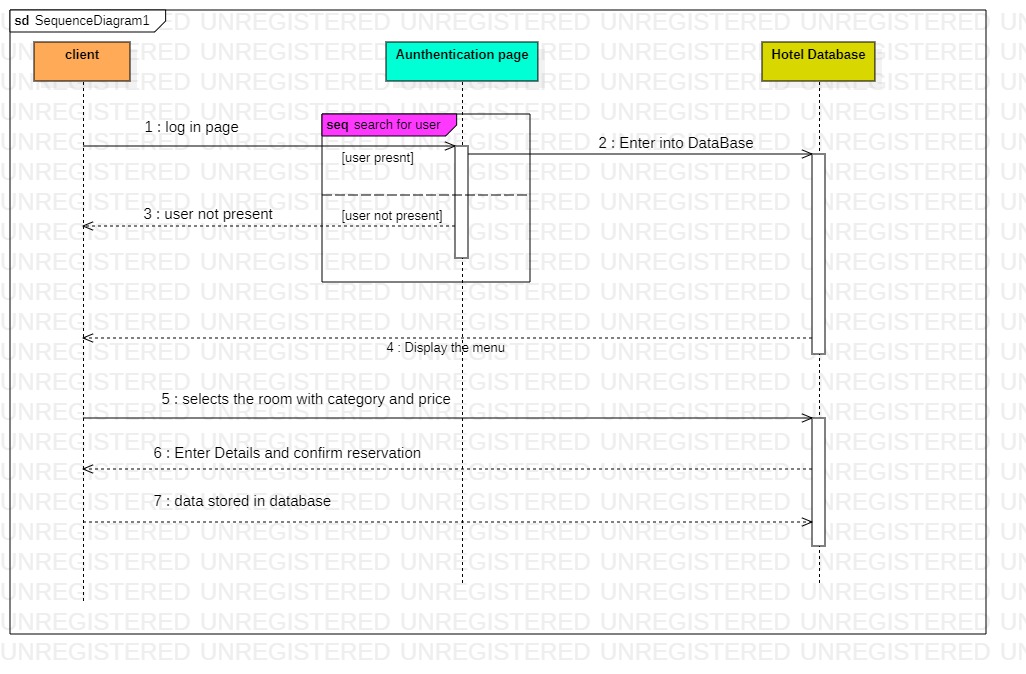
**Use-case Diagrams:**

* It is also called behavioral UML diagram. It gives a graphic over-view of the actors involved in a system directly. It shows how different functions needed by the actors and how they interact. Moreover, the diagram gives a clear picture of the user and system relationships. The below use-case diagram depicts the elements involved in the application.
* The use case diagrams show the main parts of the system and how information moves between them. The label “include” indicates that the sub-processes of the main processes which must be included to complete the task.
* The client login into the portal and they can check for conditions as mentioned in below diagram.
* This diagram shows the reservation process made by the client. The service details of the chosen reservations.
* Server has ability to update rooms availability.



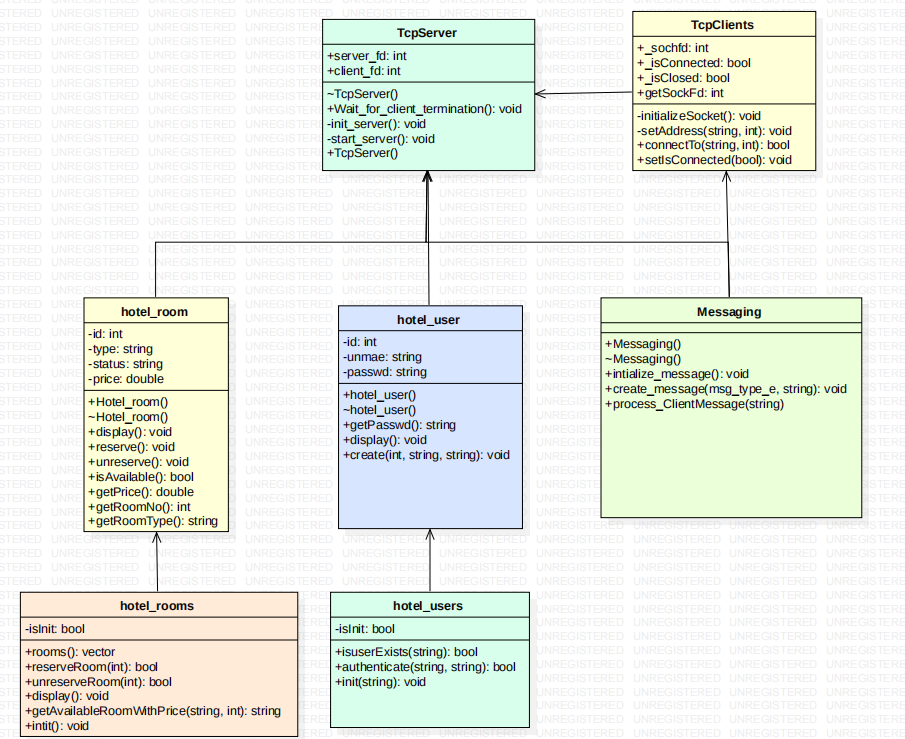
**Sequence Diagram:**

* Sequence diagram is also a type of UML which shows us the sequence of messages between two objects in a single iteration.
* In sequence diagram, Client, Authentication page and Hotel Database acts as a lifelines. Firstly, client login to the page. If the user is authenticated then it shows user is present, if not it replies to the client as user not present. User information can be stored in Hotel Database.
* Hotel database display the menu to client for selection of room with category and price. And also ask about details and confirmation of reservation.



**Class Diagram:**

This is most frequently used UML diagram in software design. It is called as a main building block of any design. It illustrates the classes in a system, attributes and operations of each class and relation between each class.

****

## Subsystem Architecture

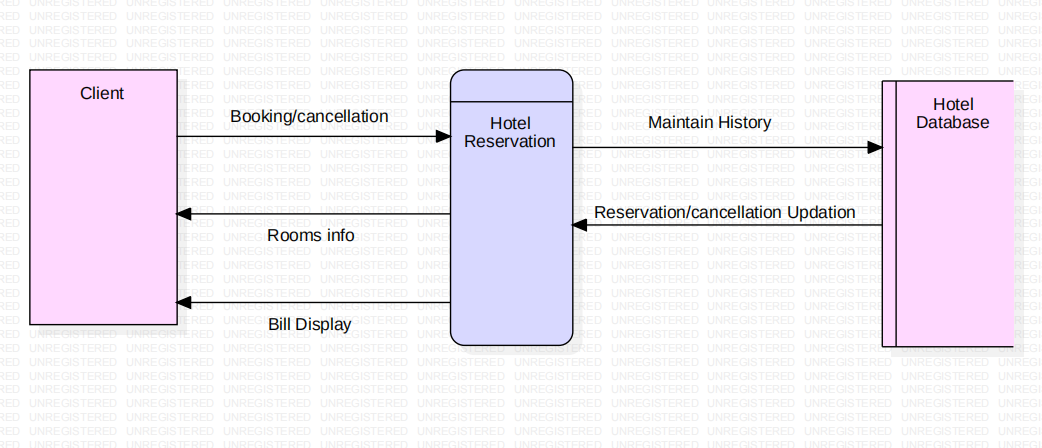
**Data flow diagram:**

Data flow diagram (DFD) is a visual aid to see the information flow within a system. It shows how data enters and leaves the system, what changes the information and flow of information, and where data is stored.

1. **0-level DFD**

The DFD Level 0 diagram for Hotel Management System is also known as the context diagram of the system. The level 0 or context diagram presents the main idea as the basis for the subsequent levels. The basic idea is represented by a single process consisting of the main process, users, and data.

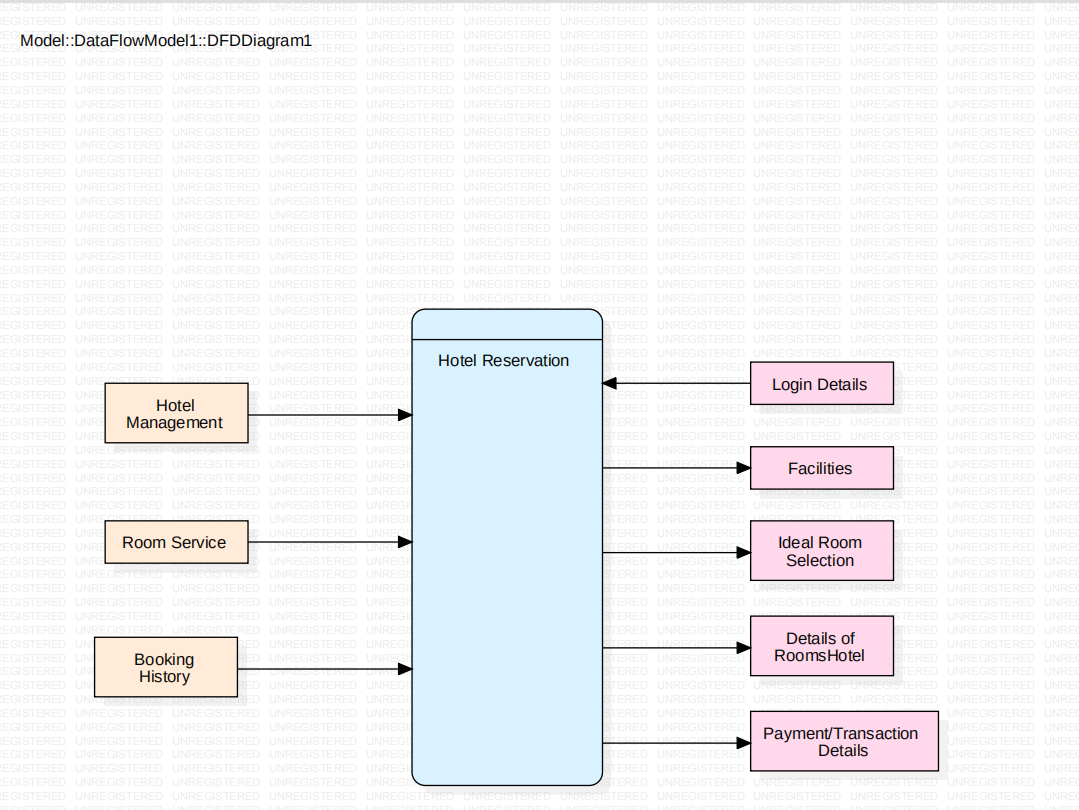
The below diagram shows the 0-level DFD of hotel Reservation system.



1. **1-level DFD**

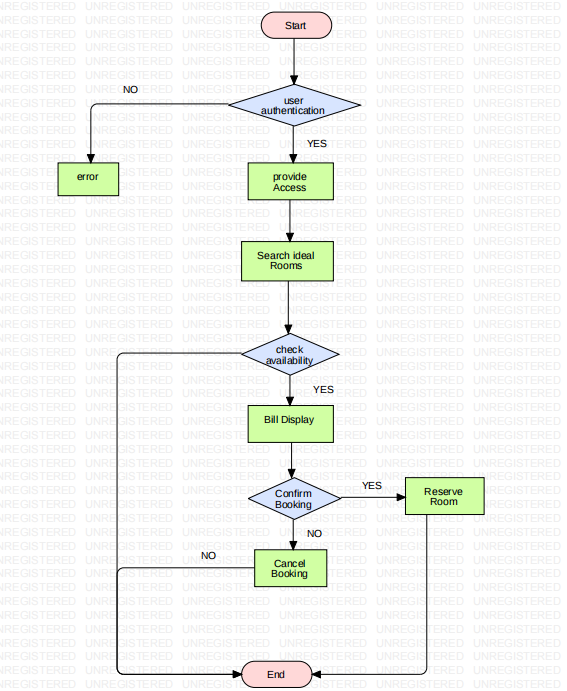
This level 1 diagram shows how data flows through the system. In addition to, its content comes from the previous level, which helps you understand the main process of the Hotel Management System even more. In DFD level 1, the diagram also shows more information about how processing works.

The below diagram shows the 1-level DFD of hotel reservation system.



**Flow chart:**

* Flow chart gives a brief description of the workflow involved in the application. The below diagram shows you the flow of work.
* In this flow chart as mentioned above, client has to go through authentication, if conditions pass they have access to search for ideal rooms. After selecting the desired rooms, it checks for room availability, it confirms the reservation from the client and proceed with the reservation.

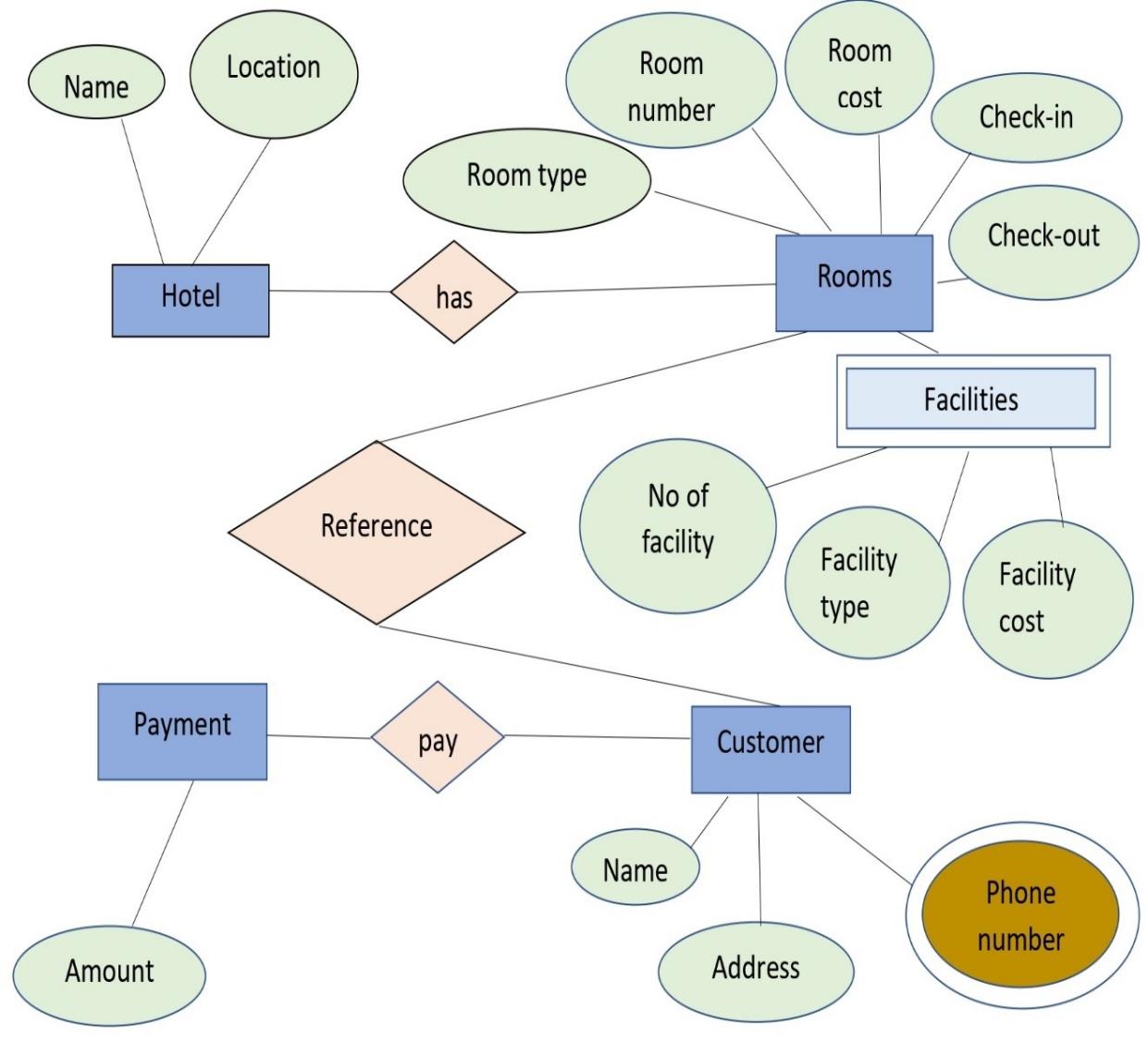


**ER Diagram:**

An Entity Relationship (ER) diagram is a type of flowchart that illustrates how entities relate to each other within a system. They use a defined set of symbols to indicate the interconnection between the entities, relationships and their attributes.

This ER diagram represents the model of Hotel Management system Entity. It shows the entire visual instrument of database tables and relationship between rooms, payments, hotel, customers etc., It used structure data and to define the relationship between structured data groups of hotel management system functionalities. The main entities of the hotel management system are hotel, rooms, services, payments, booking and customers.

The diagram below shows us the ERD of Hotel Reservation system.



## System Interfaces

Not Applicable for this project

### Internal Interfaces

Not Applicable for this project

### External Interfaces

**Hardware Interfaces**

* Necessary computer interfaces such as keyboard etc.,
* Proper Internet Connectivity

**Software Interfaces**

* Linux Based OS (capable of compiling C++ programming language)
* Server
* Database

# Detailed System Design

Here, detailed information of low-level designs are described. Low-level designs include attribute, particulars that are used while implementing application.

## Key Entities

The key entities of this project mainly include the hotel rooms, reservation, customers and hotel management. The communication between the server and client is established using TCP communication network.

## Detailed-Level Database Design

Here, we use files to store the data of registered customers and hotel information.

### Data Mapping Information

Not applicable for this project

### Data Conversion

Not applicable for this project

## Archival and retention requirements

Not applicable for this project.

## Disaster and Failure Recovery

Not applicable for this project.

## Business Process workflow

The workflow of our application mainly involves sockets in between client and server. Here client is provided with menu-based interface to select his/her requirement according to given menu.

The data which is given by client includes the type of message, length and value of the message to be sent to server. The data given by user is stored in maps STL and sent to the server. Based on the selection of client the server responds with suitable message.

## Business Process Modeling and Management (as applicable)

The business model used is Agile methodology, where the application is iterated after implementing each process in between the actual implementation of application. The steps implemented are planning phase, which involves the detailed understanding of SRS and further documenting the system requirement specification. The next step is analysing requirements and resources and then designing phase where UML diagrams are used. The next step is actual application implementation. The application is iterated for each phase of implementation.

## Business Logic

In this section, we are going to represent the important entities implemented in business logic while implementing application.

**4.7.1 authentication**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | authenticate() |  | | |
| Inutput | uname, passwd | string | NA | Client gives username and password |
| Output | none |  |  |  |
| Process | It authenticates/checks whether the user is present in the database of the server. |  | | |

**4.7.2 client\_handler**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Client\_handler |  | | |
| Input | Client\_fd | Void\* | none |  |
| Output | none | none |  |  |
| Process | It is used handles multiple clients on a server. Maximum number of clients that can access the server are defined. |  | | |

**4.7.3 reserveRoom()**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | reserveRoom() |  | | |
| Input | id | int | none | Here, id indicates the room number. |
| Output | none | none |  |  |
| Process | Used to reserve the type of room selected by the client. |  | | |

**4.7.4 unreserveRoom()**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | unreserveRoom() |  | | |
| Input | id | int | none | Here, id indicates the room number. |
| Output | none | none |  |  |
| Process | Used to cancel/terminate the reservation done by the client |  | | |

**4.7.5 getAvailableRoomWithPrice**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | getAvailableRoomWithPrice() |  | | |
| Input | type, noOfdays | string,int | none | Here, type indicates the type of room and noOfdays is the period of stay of client. |
| Output | none | none |  |  |
| Process | It gives complete details of rooms and their prices and total price multiplied with the number of days |  | | |

## Variables

1. class hotel\_users

{

public:

static bool authenticate(string uname, string passwd);

private:

static bool isInit;

};

1. class hotel\_room

{

public:

void create(int id, string type, string status, double price);

void reserve();

void unreserve();

private:

int id;

string type;

string status;

double price;

};

3) class TcpServer

{

private:

int server\_fd;

int client\_fd[MAX\_CLIENTS];

pthread\_t thread[MAX\_CLIENTS];

struct sockaddr\_in server\_addr;

static map<int,Messaging> clientMessageMap;

pthread\_t \_receiveThread;

void init\_server();

void start\_server();

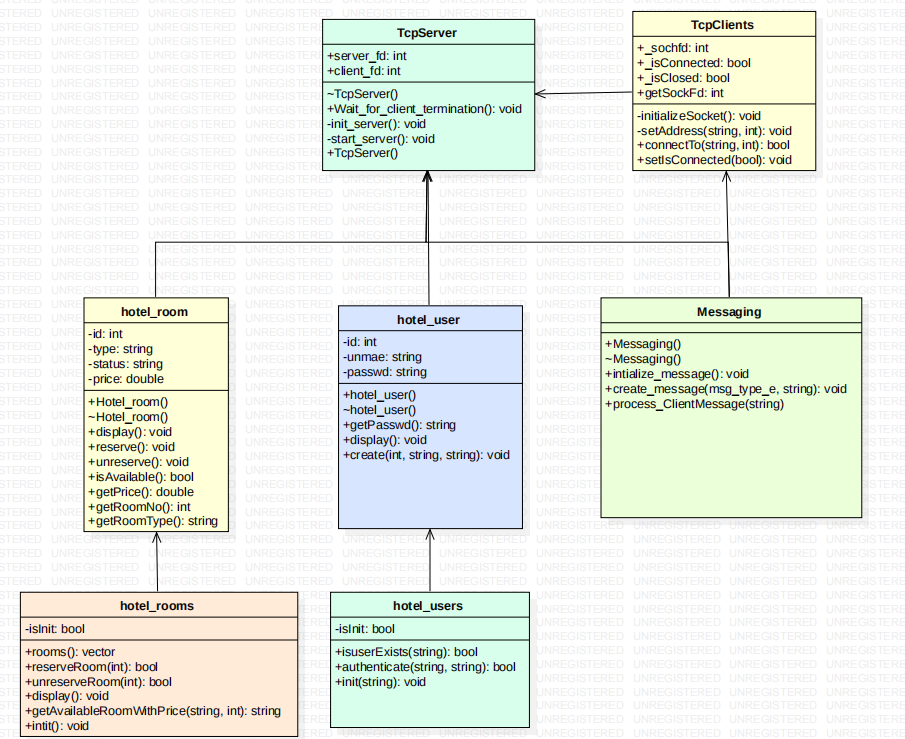
public:

static void\* client\_handler(void \*client\_fd);

void wait\_for\_client\_termination();

};

## Activity / Class Diagrams (as applicable)



Class Diagram above represents the key entities involved in implementation of the application. Here, client acts as user interface and gives the user given input to the server and server then processeses the information. After processing the information, it is then sent to client with the respective reply message. In this way, a way of communication is established between client and server.

## Data Migration

Not applicable for the project.

### Architectural Representation

Architecture representation includes use-case, sequence, dataflow diagrams of different levels. These are designed and elaborated in the respective sections accordingly.

### Architectural Goals and Constraints

The main goal of designing different UML diagrams is to understand the key entities that are to be implemented in the application and understand the workflow while implementing the application. They help us to understand the step-by-step process to be followed by the developer. The constraints are, sometimes the implemented UML doesn’t represent the system functionality perfectly.

### Logical View

### Architecturally Significant Design Packages

Not applicable to this project.

### Data model

Not applicable for this project.

**Legacy system data model**

**Proposed system data model**

**Interface data model**

### Deployment View

Not applicable for this project.

# Environment Description

This section provides all the description of the environment required by the application such as hardware requirements, software requirements, time zones etc.

## Time Zone Support

Any time zone is applicable.

## Language Support

* C++ Programming language.
* Socket programming using TCP network communication.

## User Desktop Requirements

* The cloud machine (putty) is used for developing source code.
* WinSCP is used for copying the files to local machines for a backup.

## Server-Side Requirements

* The data files require around 5MB of memory.

### Deployment Considerations

* Hardware requirements
* Software requirements
* Server requirements
* File and source code storage

### Application Server Disk Space

* As it is a small application and simple prototype of real-time application space around 5MB is applicable.

### Database Server Disk Space

Not applicable for this project.

### Integration Requirements

Not applicable for this project.

### Jobs

Not applicable for this project.

### Network

Application is developed using TCP LAN communication network.

### Others

Not applicable

## Configuration

### Operating System

Processor Intel(R) Core (TM) i5-1035G1 CPU @ 1.00GHz 1.19 GHz

System type 64-bit operating system, x64-based processor

### Database

Not applicable to this project.

### Network

It can be used on TCP oriented network.

### Desktop

Can be used on any desktop.

# References

# Appendix

**Change Log**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **QMS Template Version Control (Maintained by QA)** | | | | | |
|  |  |  |  |  |  |
| **Date** | **Version** | **Author** | | **Description** | |
|  |  |  | |  | |
|  |  |  | |  | |
|  |  |  | |  | |
|  |  |  | |  | |