REVIEW -3

SOFTWARE ARCHITECTURE AND DESIGN

SWE2004

D2

HOTEL MANAGEMENT SYSTEM

D.Harishkumar 16MIS0245

Karthikeyan K 16MIS0102

Submitted to

Prof SREE DHARINYA

HOTEL MANAGEMENT SYSTEM

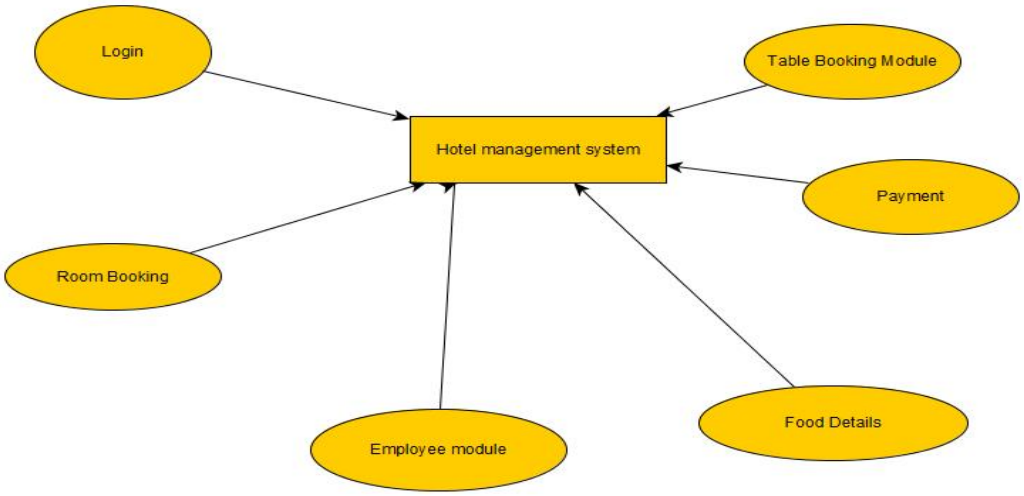
DETAILED DESCRIPTION

Travel and tourism is a booming industry today. Technology has connected cities and destinations across different countries and brought them closer. The number of people traveling across different places has increased.

Now, whether on a business trip or vacationing with family, everybody needs a place to stay and unwind. But it can be a pain in the neck sometimes getting the information about the hotels that suit your budget and then getting bookings done. Online Hotel Reservations are becoming popular method for booking hotel rooms. Travelers can book rooms from home via their home computer by using online security to protect their privacy and financial information and by using several online travel agents to compare prices and facilities at different hotels. Prior to the internet, travelers could write, telephone the hotel directly or use a travel agent to make a reservation. Nowadays, online travel agents have pictures of hotels and rooms, information on prices and deals, and even information on local resorts. Many also allow reviews of the traveler to be recorded with the online travel agent. Online reservations are also helpful for making last minute travel arrangements. Hotels may drop the price of a room if some rooms are still available. There are several websites that specialize in searches for deals on rooms. Large hotel chains typically have direct connections to the airline national distribution systems (GDS) (Sabre, Galileo, Amadeus, and Worldspan). These in turn provide hotel information directly to the hundreds of thousands of travel agents that align themselves with one of these systems. Individual hotels and small hotel chains often cannot afford the expense of these direct connections and turn to other companies to provide the connections.

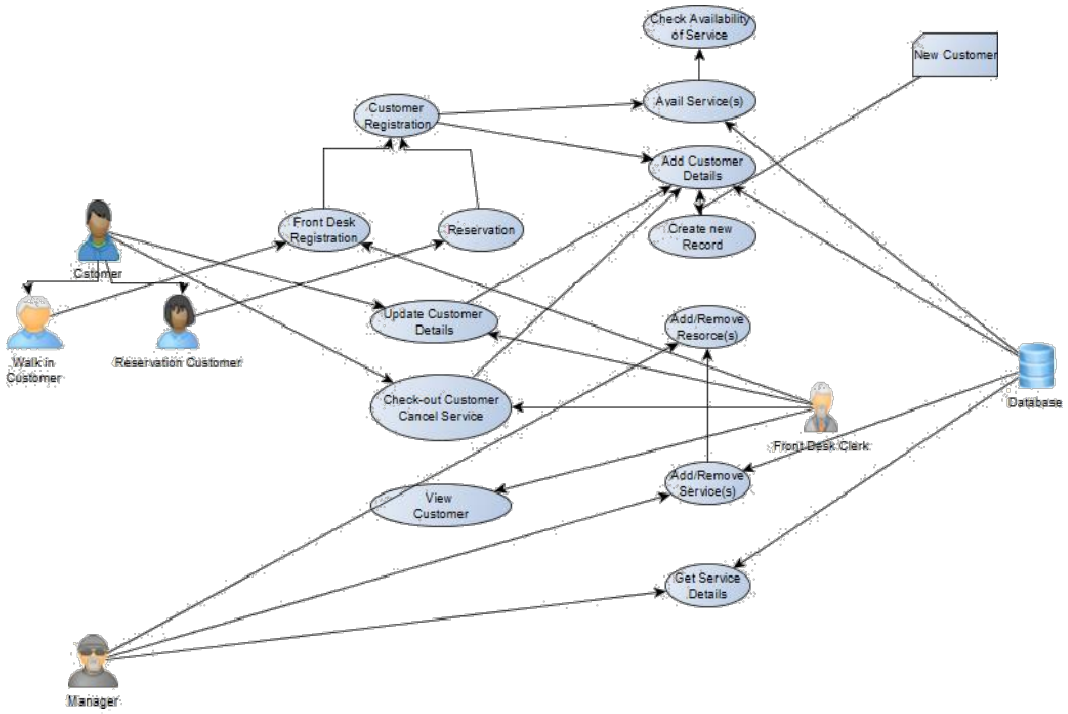
Several large online travel sites are, in effect, travel agencies. These sites send the hotels' information and rates downstream to literally thousands of online travel sites, most of which act as travel agents. They can then receive commission payments from the hotels for any business booked on their websites. People can book directly on an individual hotel's website. An increasing number of hotels are building their own websites to allow them to market their hotels directly to consumers. Non-franchise chain hotels require a "booking engine" application to be attached to their website to permit people to book rooms in real time. One advantage of booking with the hotel directly is the use of the hotel's full cancellation policy as well as not needing a deposit in most situations. To improve the likelihood of filling rooms, hotels tend to use several of the above systems. The content on many hotel reservation systems is becoming increasingly similar as more hotels sign up to all the sites. Companies thus have to either rely on specially negotiated rates with the hotels and hotel chains or trust in the influence of search engine rankings to draw in customers. The ultimate service provided by these companies to the hotels and the online consumer is that they provide a single database from which all reservation sources draw immediate room availability and rates. It is very important that hotels integrate with all the supply channels so that their guests are able to make accurate online bookings. There are many ways of making the online reservation; most of the online reservation systems use the centralized GDS system for making the reservation with the hotel directly. Examples of the GDS are Sabre, WorldSpan, Travelport. The online hotel reservation through GDS is just the tentative reservation, means that you do not need to pay at the time of reservation, instead pay at the time of check in or check out.

Overall Architectural Module

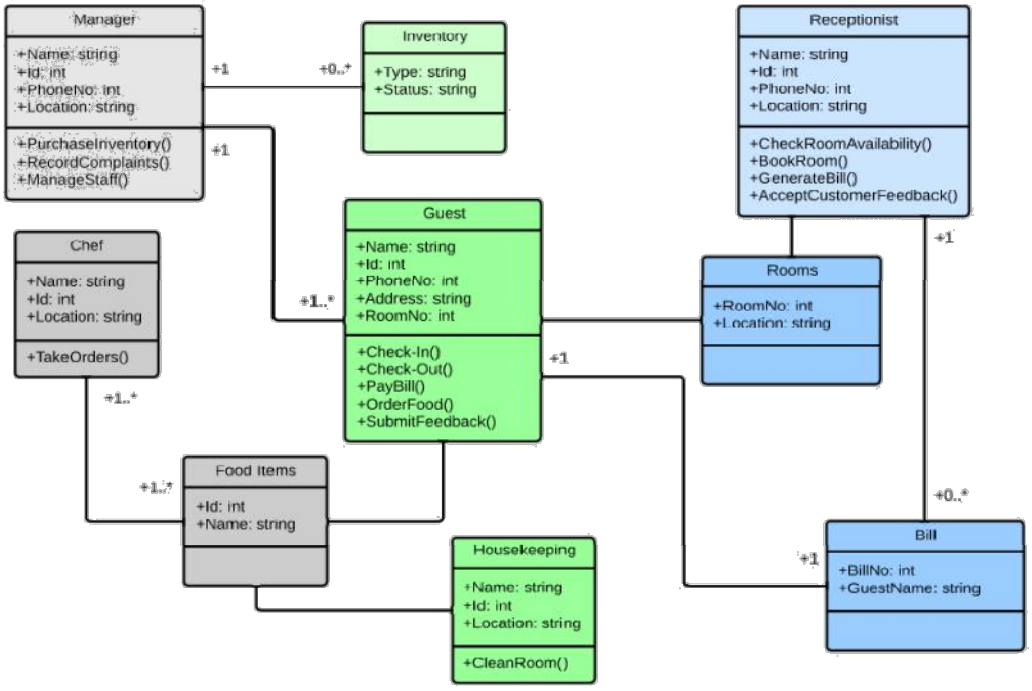


OBJECT ORIENTED DESIGN

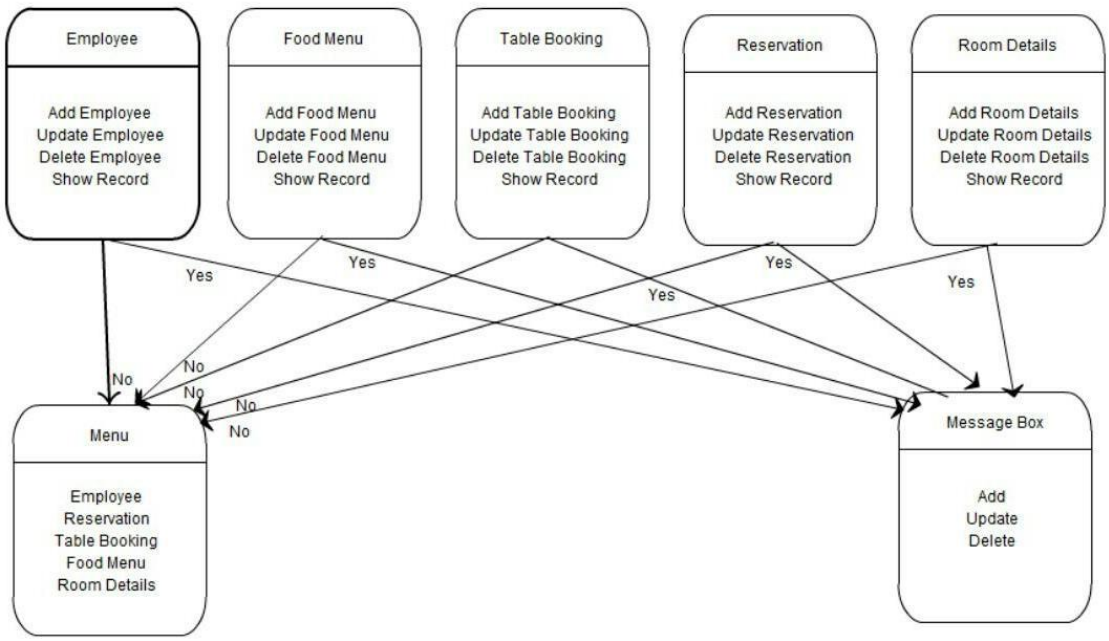
USECASE



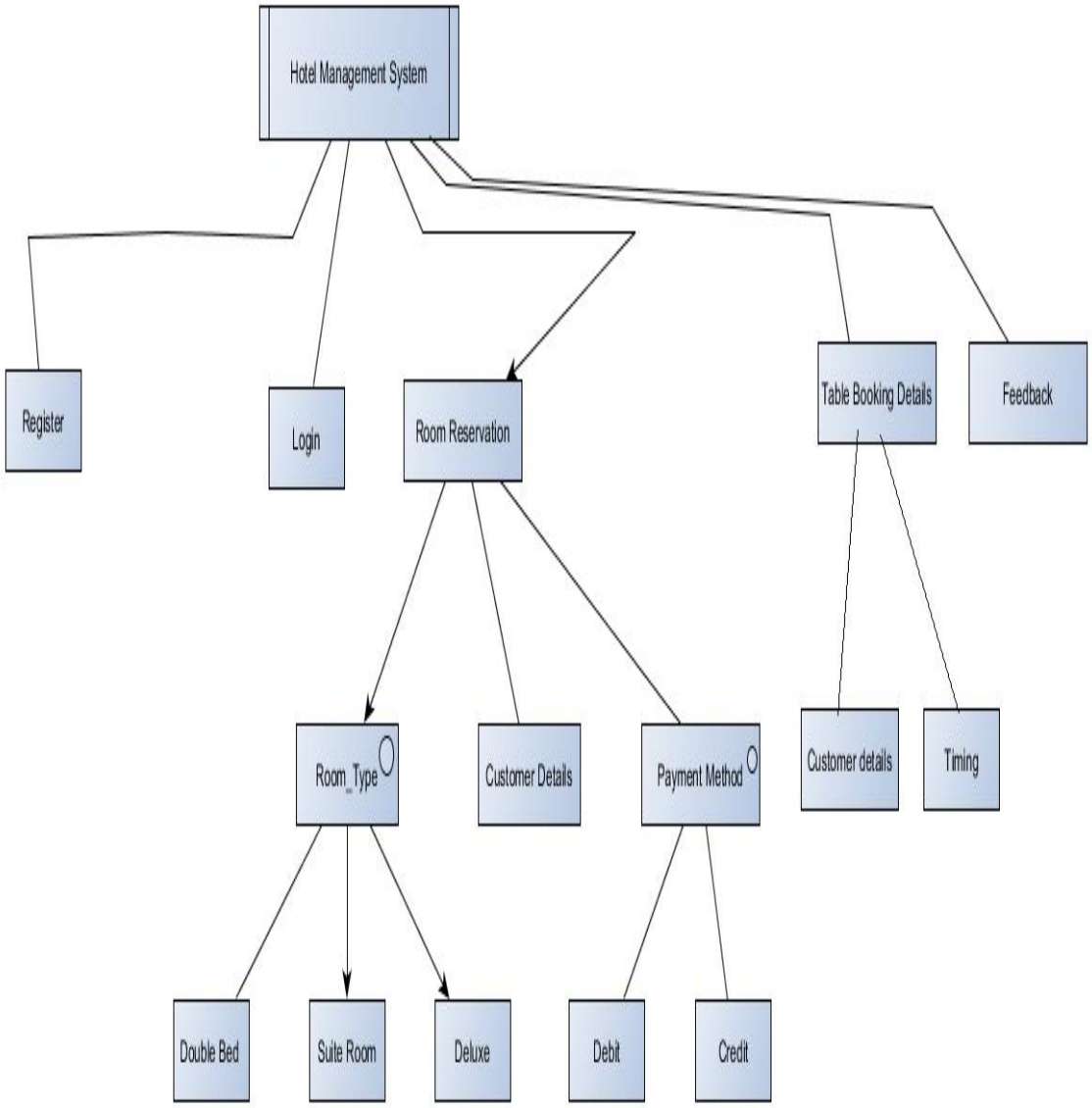
CLASS DIAGRAM


Manager
+Name: string
+Id: int
+PhoneNo: int
+Location: string
+PurchaseInventory()
+RecordComplaints()
+ManageStaff()
Inventory
+Type: string
+Status: string
Receptionist
+Name: string
+Id: int
+PhoneNo: int
+Location: string
+CheckRoomAvailability()
+BookRoom()
+GenerateBill()
+AcceptCustomerFeedback()
Guest
+Name: string
+Id: int
+PhoneNo: int
+Address: string
+RoomNo: int
+Check-In()
+Check-Out()
+PayBill()
+OrderFood()
+SubmitFeedback()
Chef
+Name: string
+Id: int
+Location: string
+TakeOrders()
Rooms
+RoomNo: int
+Location: string
Food Items
+Id: int
+Name: string
Housekeeping
+Name: string
+Id: int
+Location: string
+CleanRoom()
Bill
+BillNo: int
+GuestName: string
+1
+0..*
+1
+1..*
+1
+1..*
+1..*
+1
+1
+0..*
+1

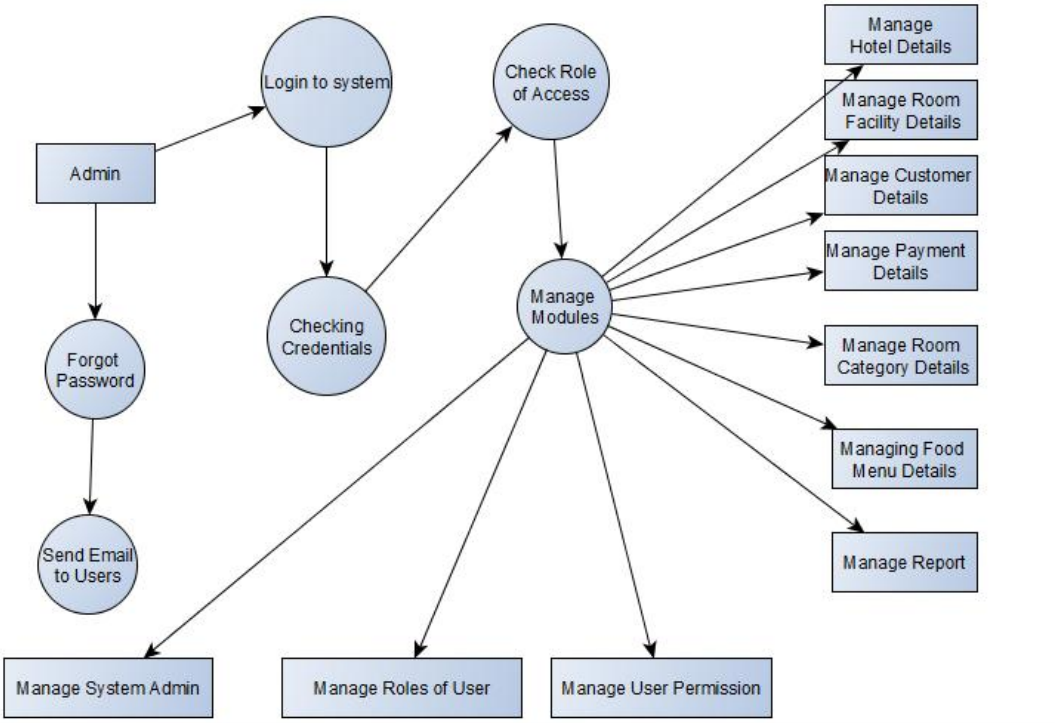
State chart diagram



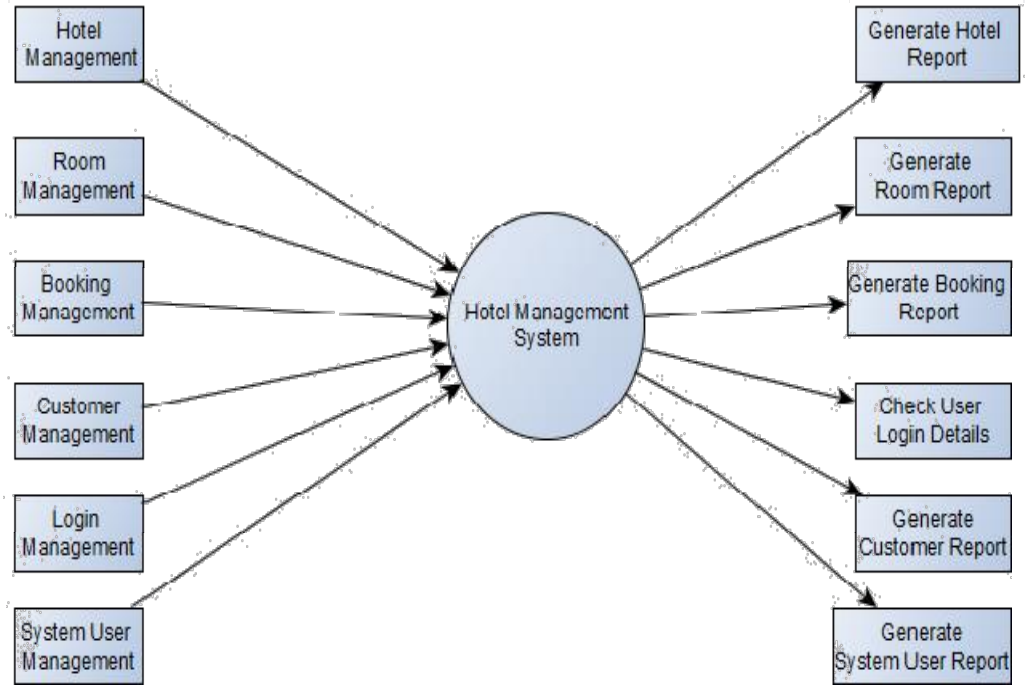
Structure Diagram using JSD



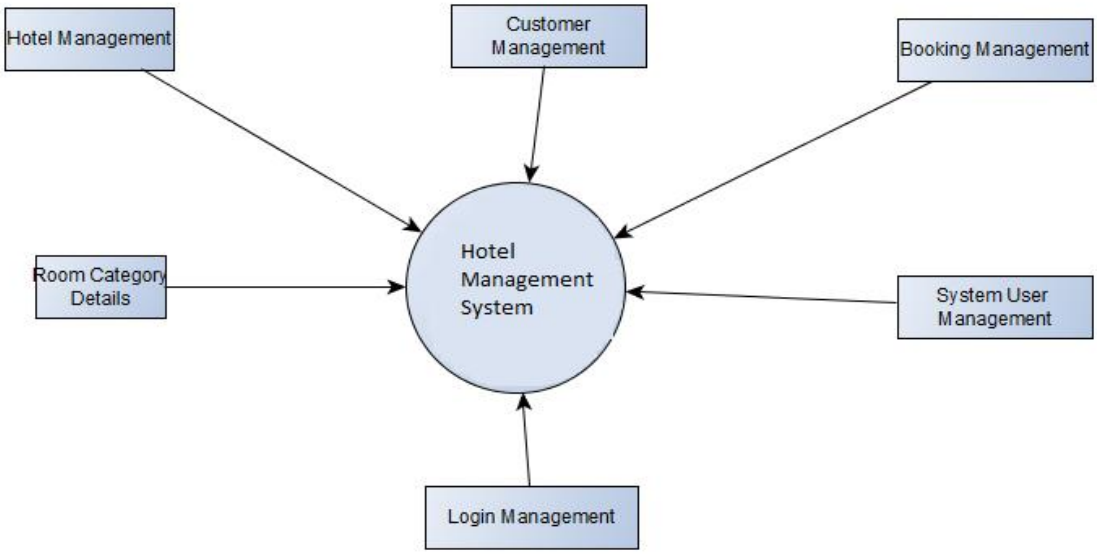
Transaction Analysis of SSAD



LEVEL 2 DFD



LEVEL 1 DFD



LEVEL 0 DFD

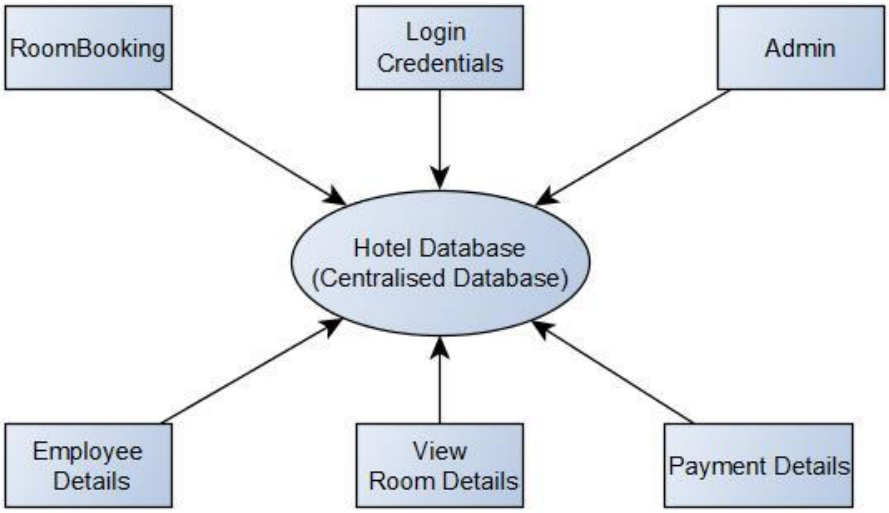
Architecture styles

- Implicit invocation
- Layering
- Repositories
- Interpreters
- Process control
- Pipes and filters
- Objects(Data Abstraction & Object oriented Organization)

Relating Architectural Style with the modules of Hotel Management System

REPOSITORY MODEL

A database for the Whole System for storing details like login credentials, payment details, making reservations for table, booked rooms is considered **Centralized repository model**



Components:

The Components are Admin, Login Credentials, Admin, Employee Details , View Room Details, Payment Details.

Connectors :

The Connectors are used to connect the Database with all the modules which are mentioned in above diagram.

Topology :

The layers of data that can read all other modules from one module

Semantic constraints :

The layers in cooperate with the other layers just beside were the sematic constraint's.

ADVANTAGES :

Each and every module like Login Credentials, Admin, Employee Details , View Room Details are independent.So, it is easy to add other modules into it

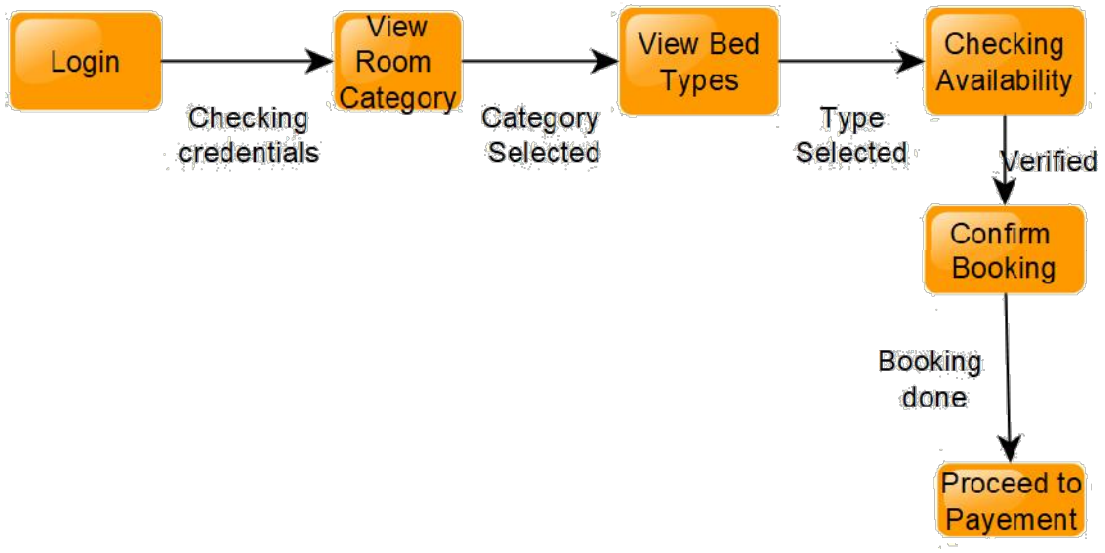
DISADVANTAGES :

- Security is main issue here as every module incorporates with the central model
- Availability of the software was also a problem here as it was go shut down time by time

PIPE AND FILTER ARCHITECTURE

Pipe and Filter Architecture is used for Room booking module. Because, it contains several components in order to complete the room booking task.

Room Booking Module



FILTER :Actually the filter is used to provide the functionality for booking the Room

If the data user name and password were correct then only it allows the user to enter into the system

PIPE :Here pipes were used to move the data from one module to the other modules

TOPOLOGY :the layers of filters were considered as topology (the data that was flowed from start to end will have connection between them [layers])

SEMANTIC CONSTRIANTS :as one filter contains the relation with pipe that was either side so they were call sematic constraint's

Components:

Components are filters here

- Login
  - View Room Category
  - View Bed types
- Checking Availability

Connectors:

Connectors were pipes that take the data into forward direction

- Checking credentials
- Type Selected

Verification
Booking Done

Description :

Here the user will give input as user name and password

filters will verify the data and wrongly encrypted data will be stopped or

halted and pipes were used to take the data forward into different modules

First filter checks the

username pipe moves data

Second filter checks the password

pipe moves the data

Third filter gets the category

Pipe moves the data

Fourth filter gets bed type

Pipe moves the data

Fourth filter checks the availability

Pipe moves the data

This is how the pipe and filter was used in the login form of the hotel management system

.

ADVANTAGES

- designer to understand
- support reuse
- easily maintained and enhanced
- specialized analysis

It is easy for the drive to understand the design

As we all know that it will support reuse

How ever login form can be easily enhanced and maintained

DISADVANTAGES :

Piper and filter allows batch sequential

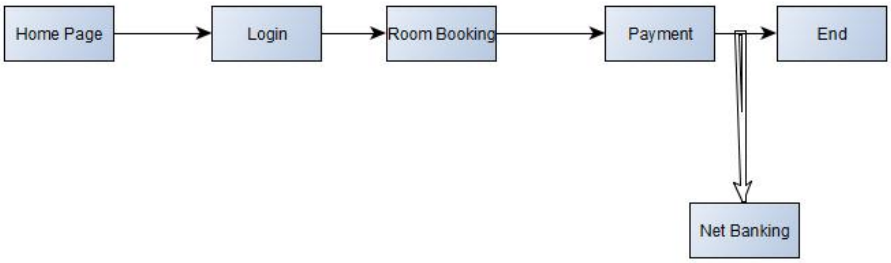
That means the coordination between one filter and other filter was very less

Usually the login forms made of pipe and filter architecture were very low in performance .

PROCESS CONTROL ARCHITECTURE STYLE

Process Control Architecture style can be used for Payment Module in Hotel Management System. Because, in these module , the process is controlled by both bank and the developer.

In these module, the link leads to the different net-banking website which is developed and maintained by different banking organisation. Here the other system and bank will control the payment process



COMPONENTS :

The components were the various activities that were going in a module

Here in payment portal the bank, other website for payment will act as the components

Connectors :

The process that invokes between these various components were connectors

Topology :the bunch of different kinds of components like bank and other websites are called as topology

sematic constraints :

The link between the process like reserve and between bank was not there but were in the same components .

ADVANTAGES :

When ever there is continuous process then the system od process control was used

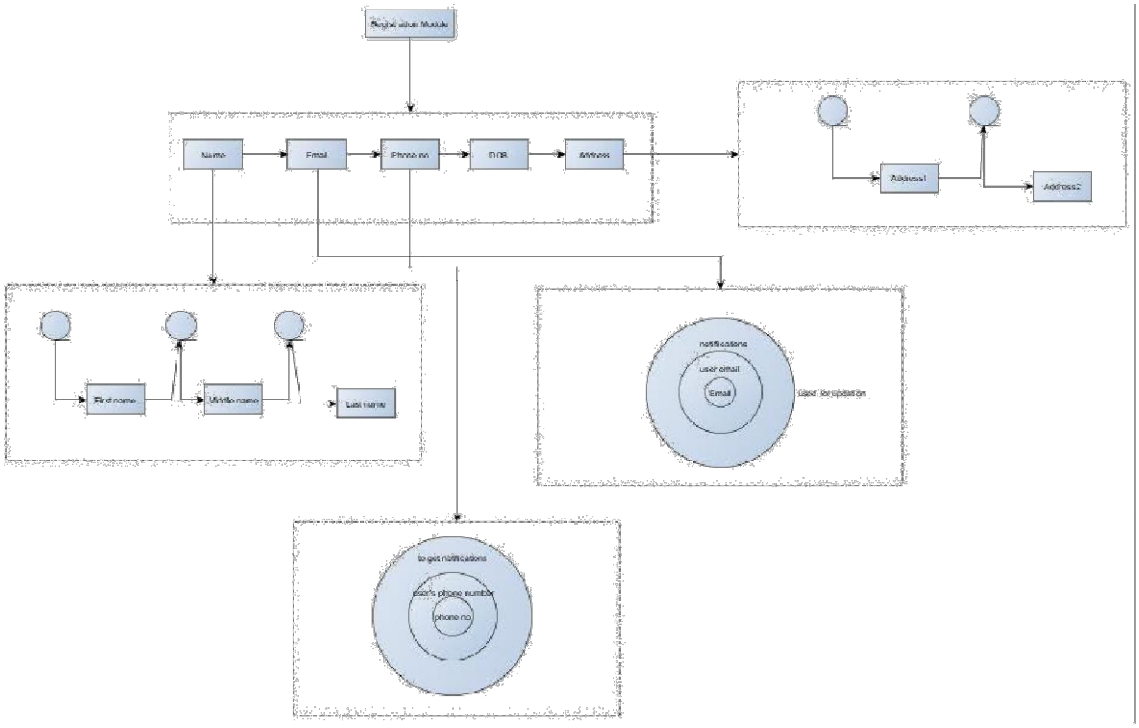
Like here in the reserving ticket ,cancelling like that

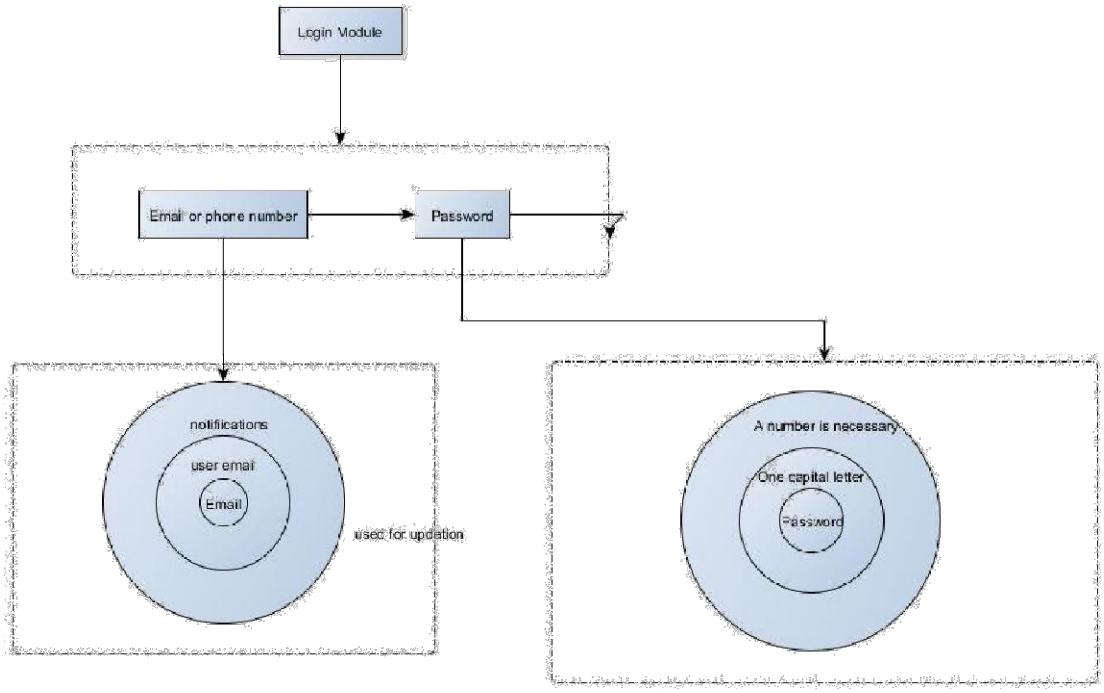
DISADVANTAGES :

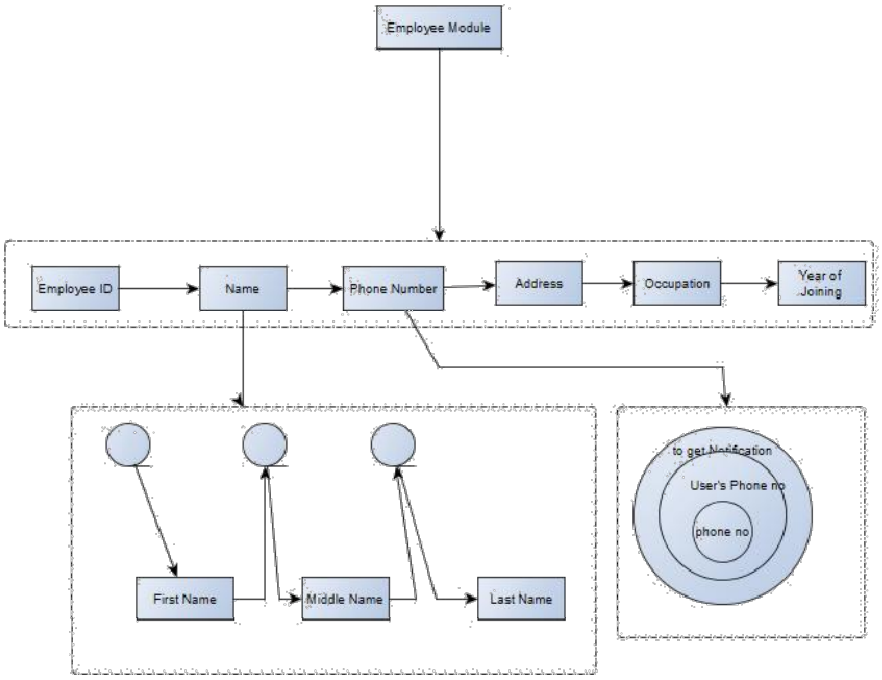
If they are many interaction process like bank ,otp,organization etc etc then it cant be incorporate all those things .

LAYERED MODEL(LAYERING)

For Hotel Management system, we use Layered Architecture for registration module, login module, and Employee module


Name
Email
Phone no
DOB
Address
Address1
Address2
First name
Middle name
Last name
notifications
user email
Email
used for updation
to get notifications
user's phone number
phone no


Login Module
Email or phone number
Password
notifications
user email
Email
used for updation
A number is necessary
One capital letter
Password



For registration module, we get the mode of payment like login credentials for net banking, cash ondelivery, and debit and credit cards

Here the layer of activities were happening one by one depending on each other

COMPONENTS :the sub programs like login, showing the train status were called as the components .

CONNECTORS :the action that were going to take place in between the components was called as the connectors .

ADVANTAGES :

First, they support design based on increasing levels of abstraction, as we can see that the hierarchy level is followed here in abstraction

• Second, they support enhancement, they are very easy to understood

• Third, they support reuse ,they can be modified in any manner and can be reused

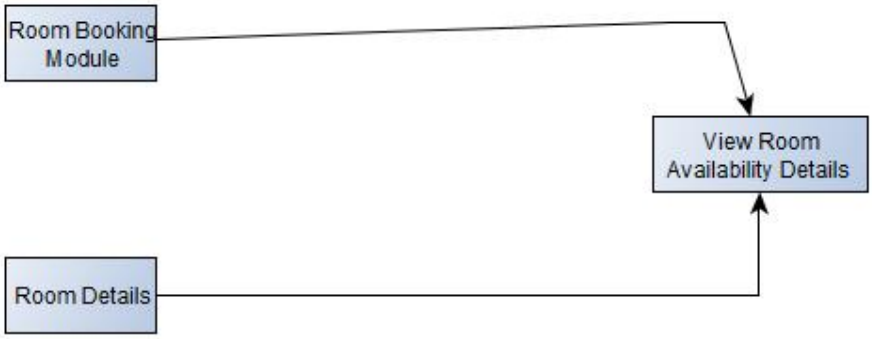
DISADVANTAGES :

- Every system was not dependent on other so it was difficult to use a layered architecture style .As not many modules of train reservation was not included
- Very hard to find the systems that were dependent on each other.

IMPLICIT INVOCATION

Implicit Invocation is used “Room Availability Details“

For getting the Room Availability Details for a Hotel , one should invoke the Room Booking Module and Room Details to know the correct status by the user in order to book the room. This is how it was implemented



COMPONENT :Component is a event that is going to be happen ,here the innovation of room details is a component

CONNECTORS :

Connectors were the registered process, as room availability status was registered with view room details it was considered as the connectors

Topology : the data which was that was present in the room availability status

Sematic constrainsts :different data in invocation will call different procedures of room details each may not be depenendent .

Here In the above diagram the view room availability status was registered with train details and reserve room module

When ever the user gives the input that will invocate that module to get the details .

Description:

Here the event that is view room availability status was registered with other modules and invocates the output whenever it was needed /an input is received from the user.

Advantages :

The concept of reusability is used here ,here the same can be incorporated in many ways and be reused

They are distributed equally so the independent of each component was clear here

Any input can be viewed and used so the visibility was high any user can get the data needed.

Disadvantages :

The components way give reply or not the possibility is too low

As if the view room availability status was registered with reserve room we cant say that for every item (for every event) the output was invoked as details

-----so reliabity is low

The complexity is very high as giving input ,output ,registering all those were difficult to understadibility of the user will be low in this case

ARCHITECTURAL MAPPING USING DATA FLOW | TRANSFOEM MAPPING

Transform mapping is a set of design steps that allows a DFD with transform flow characteristics to be mapped into a specific architecture style

To illustrate this approach ,we again consider the example of hotel management system

DESIGN STEPS :

The design steps for transaction mapping are similar and in some cases identical to steps for transform mapping . A major difference lies in the mapping of DFD to software structure.

Step 1. Review the fundamental system model.

Step 2. Review and refine data flow diagrams for the software.

Step 3. Determine whether the DFD has transform or transaction flow characteristics.

Step 4. Identify the transaction centre and the flow characteristics along each of the action paths.

Step 5. Map the DFD in a program structure amenable to transaction processing.

Step 6. Factor and refine the transaction structure and the structure of each action path.

Step 7. Refine the first-iteration architecture using design heuristics for improved software quality.

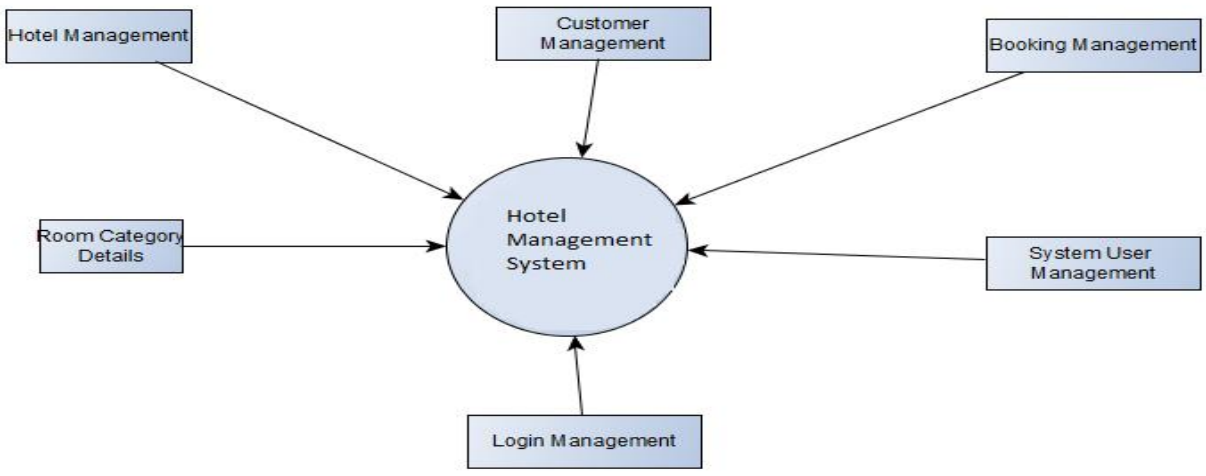
The above steps helps us to form the transform mapping

Let's see the transaction Mapping for Hotel management system.

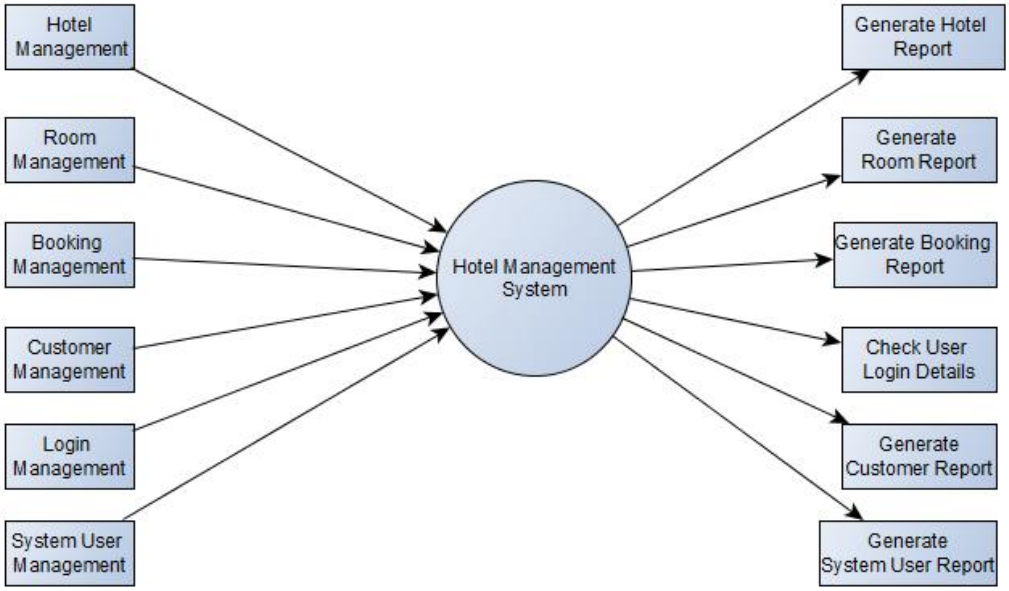
At first drawing the dataflow diagram for the Hotel management system

Level 0 DFD

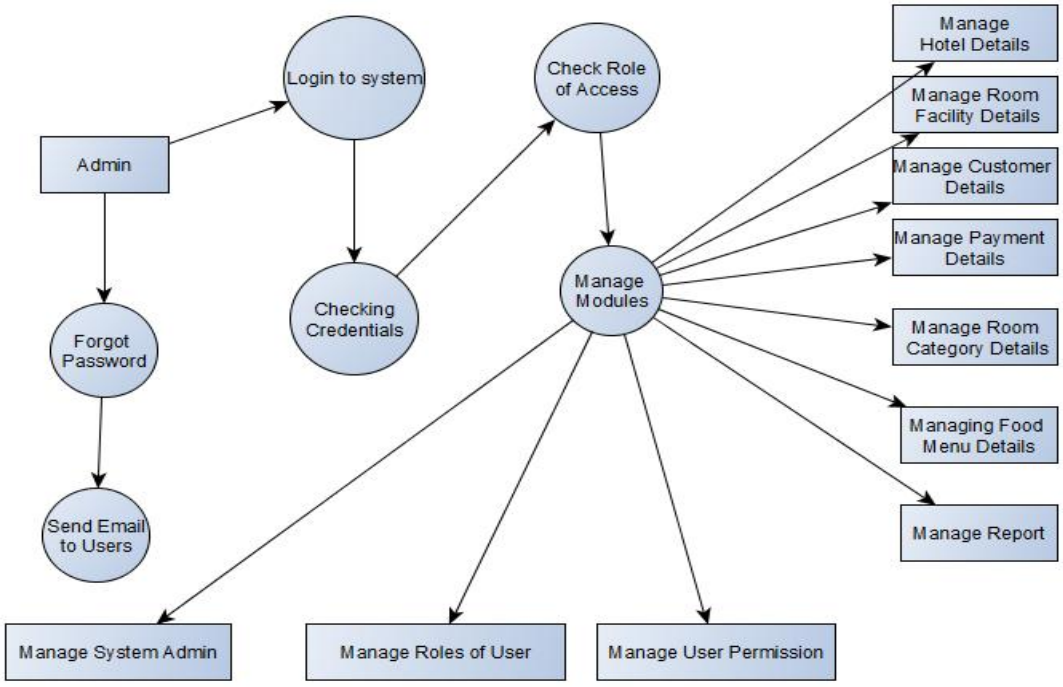
Hotel management system



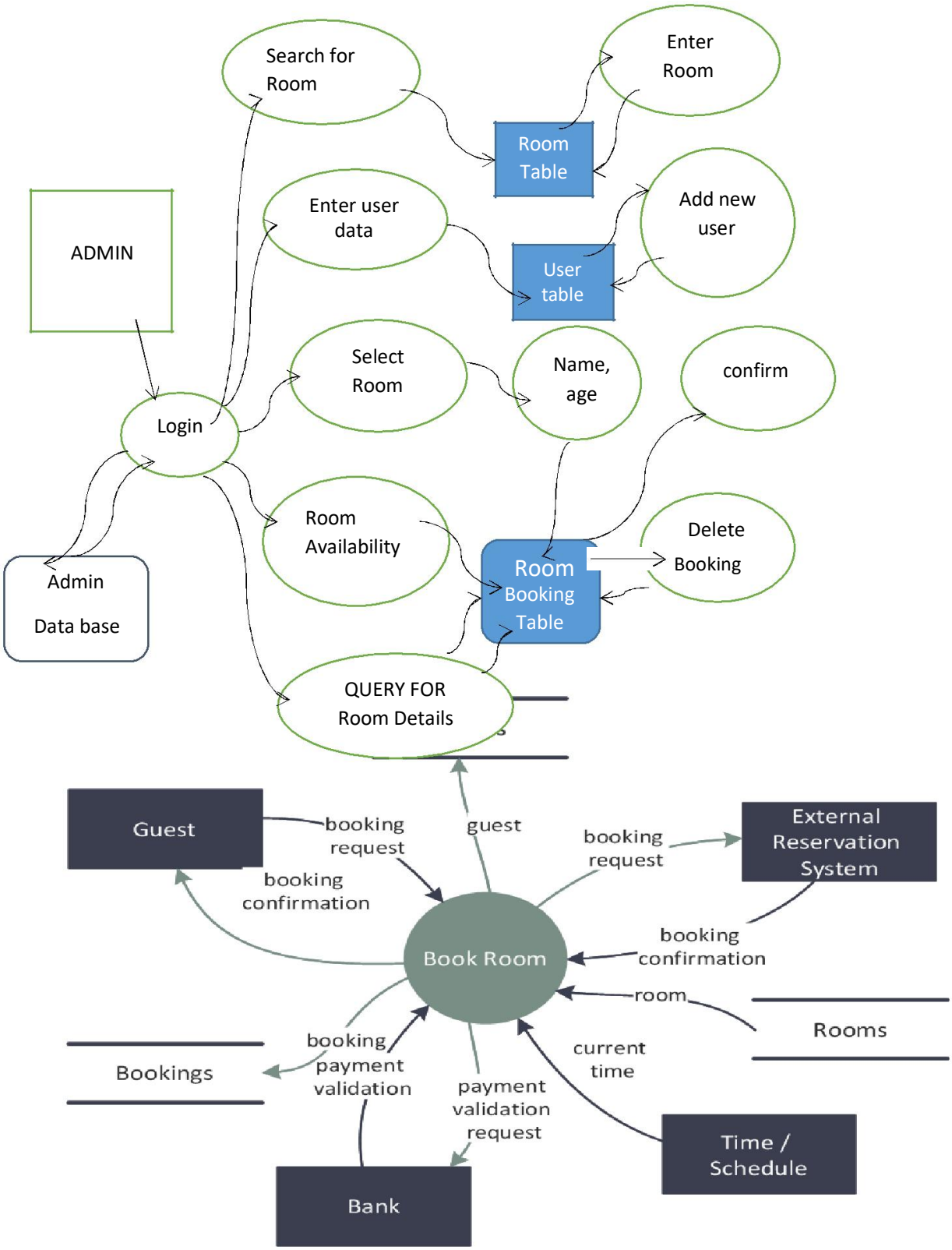
LEVEL 1 DFD :



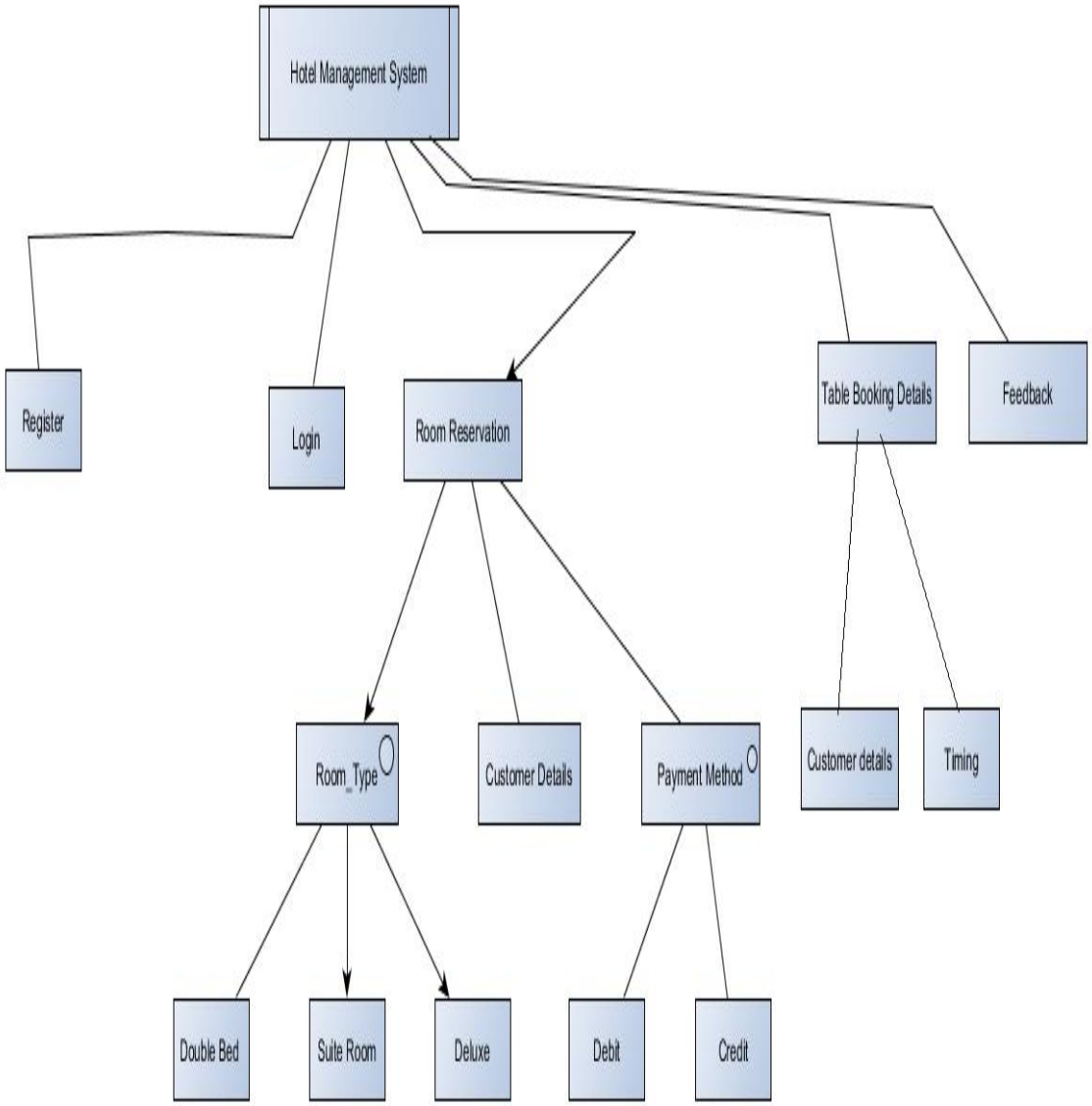
LEVEL 2 DFD :



NOW PERFORMING TRANSACTION MAPPING ON LEVEL 2 DFD



Performing step wise panning for the mapping :



Therefore the neat mapping was done and explained

ARCHITECTURAL DESIGN PATTERNS:

An architectural pattern is a general, reusable solution to a commonly occurring problem in software architecture within a given context.[1] Architectural patterns are similar to software design pattern but have a broader scope. The architectural patterns address various issues in software engineering, such as computer hardware performance limitations, high availability and minimization of a business risk. Some architectural patterns have been implemented within software frameworks.

I will be briefly explaining the following 10 common architectural patterns with their usage, pros and cons.

1. **Layered pattern**
2. **Client-server pattern**
3. **Master-slave pattern**
4. **Pipe-filter pattern**
5. **Broker pattern**
6. **Peer-to-peer pattern**
7. **Event-bus pattern**
8. **Model-view-controller pattern**
9. **Blackboard pattern**
10. **Interpreter pattern**

A brief explanation of each pattern :

Layered pattern

This pattern can be used to structure programs that can be decomposed into groups of subtasks, each of which is at a particular level of abstraction. Each layer provides services to the next higher layer.The most commonly found 4 layers of a general information system are as follows.

Presentation layer (also known as **UI layer**)

Application layer (also known as **service layer**)

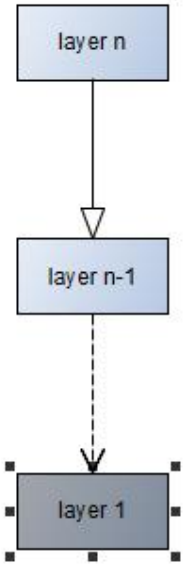
Business logic layer (also known as **domain layer**)

Data access layer (also known as **persistence layer**)

Usage

General desktop applications.

E commerce web applications.



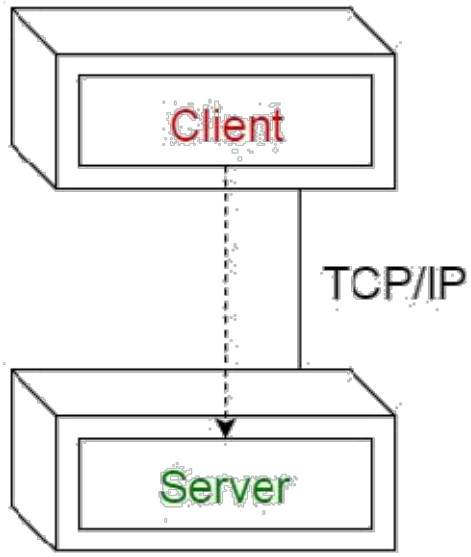
Layered pattern

Client-server pattern

This pattern consists of two parties; a **server** and multiple **clients**. The server component will provide services to multiple client components. Clients request services from the server and the server provides relevant services to those clients. Furthermore, the server continues to listen to client requests.

Usage

Online applications such as email, document sharing and banking.



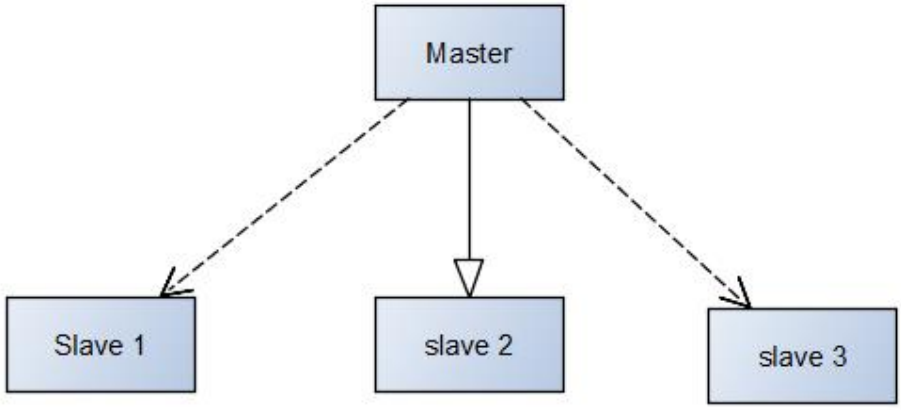
Client-server pattern

Master – Slave pattern

• Master-slave pattern supports fault tolerance and parallel computation.

• The master component distributes the work among identical slave components

• and computes a final result from the results the slaves return.



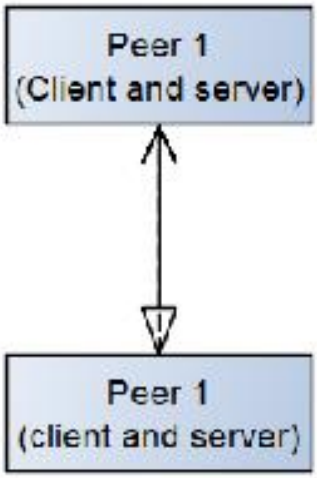
Pipe – Filter pattern :

Provides a structure for systems that produce a stream of data.

Peer-to-peer pattern

• The Peer-to-peer pattern can be seen as a symmetric Client-server pattern.

• peers may function both as a client, requesting services from other peers, and as a server, providing services to other peers.



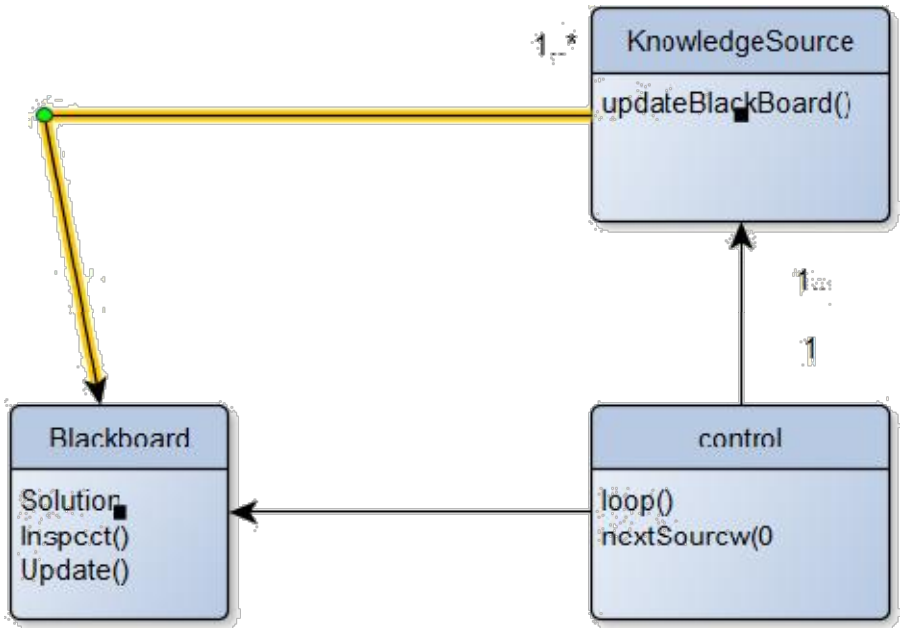
Blackboard pattern:

• The Blackboard pattern is useful for problems for which no deterministic solution strategies are known.

• All components have access to a shared data store, the blackboard.

• Components may produce new data objects that are added to the blackboard

. • Components look for particular kinds of data on the blackboard, and may find these by pattern matching.



Interpreter pattern

• The Interpreter pattern is used for designing a component that interprets programs written in a dedicated language.

HERE WE WILL SEE ANY ARCHITECTURE PATTERNS FOR ANY TWO MODULES :

HERE AS MENTIONED IN THE PROBLEM STATEMENT

We can use the VIEW PNR STATUS module for **model view controller**

1) MODEL VIEW CONTROLLER :

Here model view controller pattern was used for the VIEW PNR STATUS

But how?

This pattern, also known as MVC pattern, divides an interactive application in to 3 parts as,

1. **model**—contains the core functionality and data

2. **view**—displays the information to the user (more than one view may be defined)
3. **controller**—handles the input from the user

Here viewing the PNR status is the main core that is the ***MODEL*** functionality and data

VIEW will show all the data of respective trains to the user ,it enables the view of all other train details

CONTOLLER is responsible for hadiling the input data that is the train number and other details that were going to be entered by the user in a particular module

Here controller filters the data in all the aspects so the unwanted data will also be abstracted .

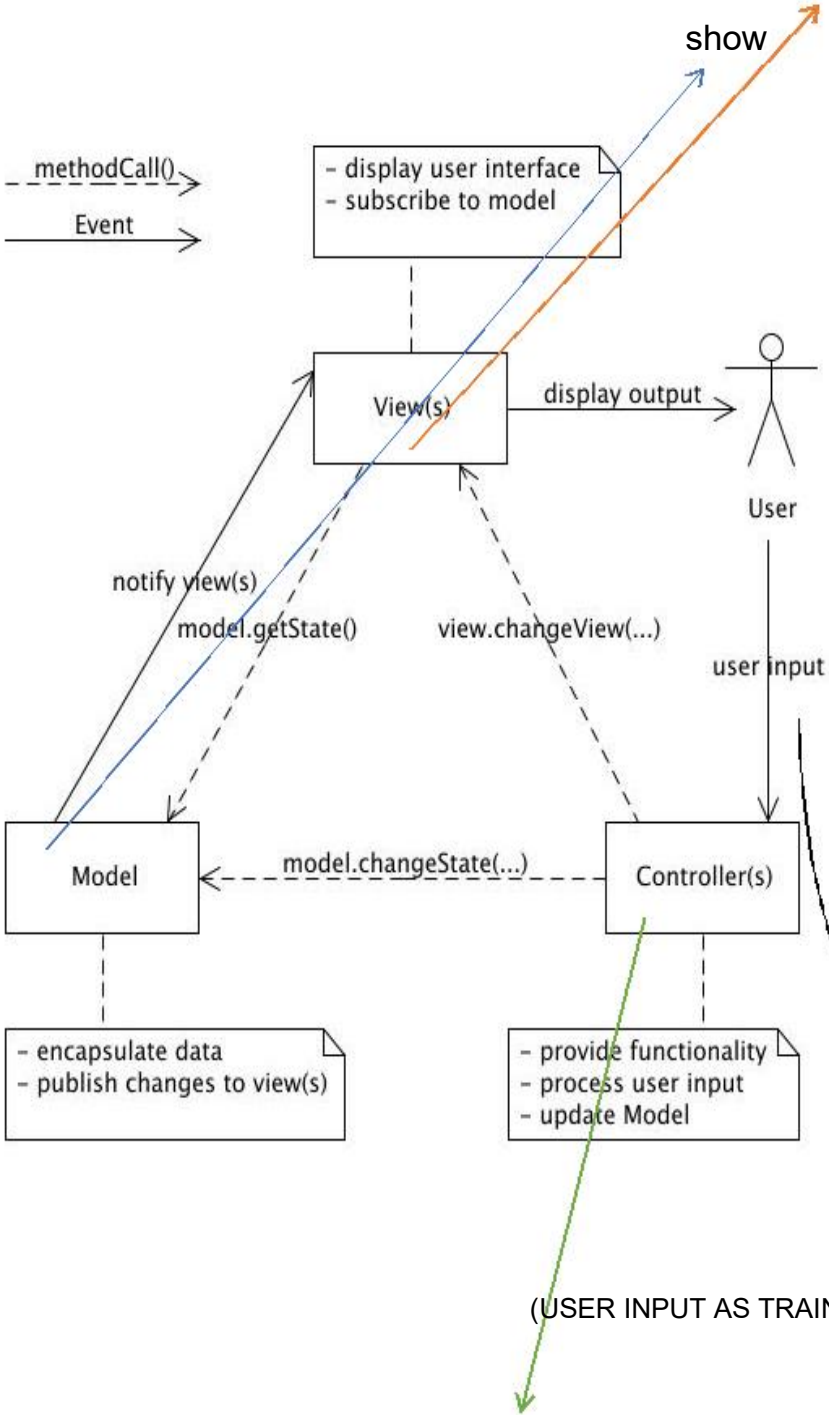
View shows the data to the user that was filtered, abstracted and modified data.

FOR THE MODULE OF VIEW PNR STATUS :

encapsulate the data and it contains all the data of trains

Filter the data and limitate the information

show to user



(USER INPUT AS TRAIN NUMBER,DESTINATION

GIVES THE DATA A PROPER FORM TO FETCH FROM MODEL

Allows only the particular data to Model to fetch that (only particular room number)

ADVANTAGES

Makes it easy to have multiple views of the same model, which can be connected and disconnected at run-time

DISADVANTAGES

Increase in the complexity makes unwanted user updates

∴ view room availability status module with MODEL VIEW CONTROLLER PATTREN was explained neatly with diagram's (go in depth to understand the concept behind) and see how it was explained

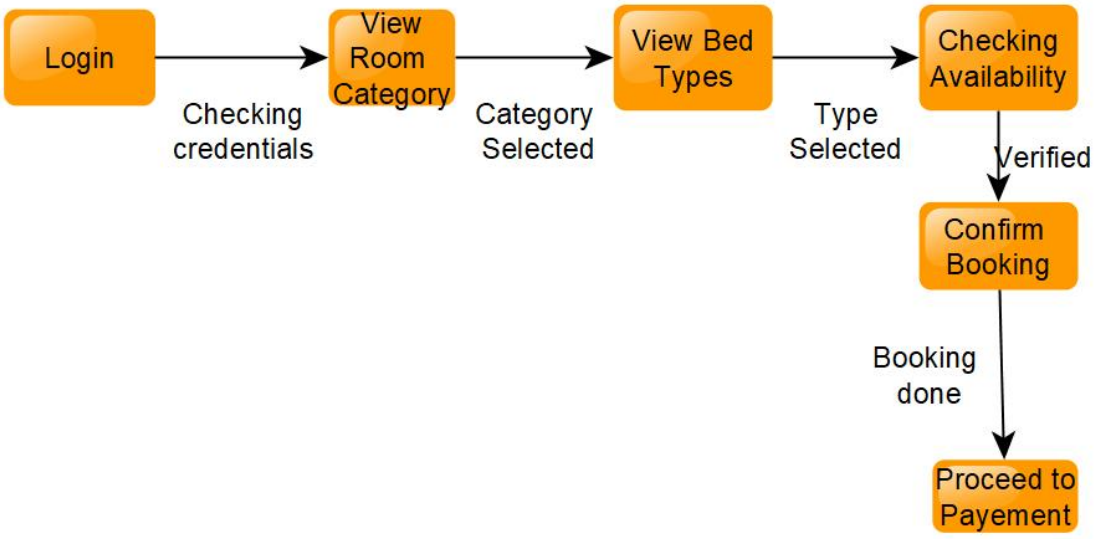
2)PIPE AND FILTER PATTREN

here the pipe and filter pattern was used in the login module

BUT HOW??

They are multiple accounts in which the users can be logging in we want to control the entire flow of that data for that purpose pipe and filter pattern in used

This pattern can be used to structure systems which produce and process a stream of data. Each processing step is enclosed within a **filter** component. Data to be processed is passed through **pipes**. These pipes can be used for buffering or for synchronization purposes.



Here the filter checks the user input data if password and user name was correct then It allows the user to enter into the next module

This is how pipe and filter pattern was used in the login page of hotel management system .

ADVANTAGES:

Exhibits concurrent processing .when input and output consist of streams, and filters start computing when they receive data .

Easy to add filters .the system can be extended easily

Filters are reusable ,can build different pipelines by recombining a given set of filters

DISADVANTAGES :

Efficiency is limited by the slowest filter process

Data transformation overhead when moving from one filter to the other

∴ **PIPE AND FILTER PATTERN** was explained neatly with the necessary modules and diagrams

3) EVENT BUS PATTREN

This pattern primarily deals with events and has 4 major components; **event source**, **event listener**, **channel** and **event bus**. Sources publish messages to particular channels on an event bus. Listeners subscribe to particular channels. Listeners are notified of messages that are published to a channel to which they have subscribed before.

Here in our problem statement event bus pattern will be used in the

PAYMENT MODULE

In Payment module we come across,

1)specific bank

2)hotel room booking website

These two sources will be publishing the 2 different channels into the bus

That two **DIFFERENT CHANNELS** were

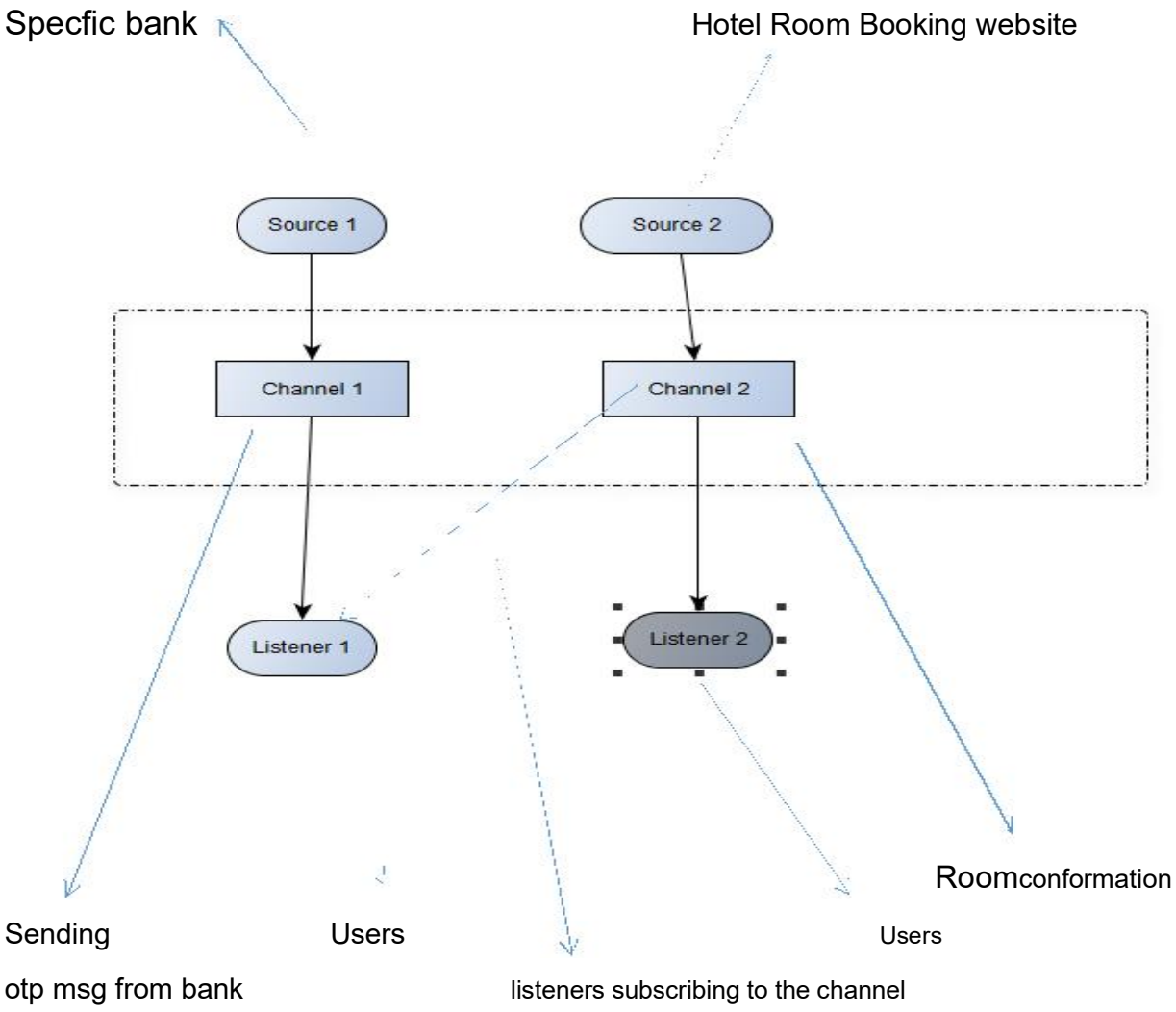
1)sending otp msg from bank

2)Room reservation and room number as a message for conformation

Then ,

Listeners are notified of messages that are published to a channel to which they have subscribed before.

Here the users have subscribed the channel before so the sms will be sent to only those members who have registered



ADVANTAGES :

New publishers ,subscribers and connections can be added easily effectively for highly distributed appications

DISADVANTAGES:

Scalability may be a problem ,as all messages travel through the same vent bus

∴ Event bus pattern was explained neatly with the example of payment module in hotel management system.

THEREFORE A NEAT MAPPING OF DESIGN WITH ARCHITECTURE AND ARCHITECTURE PATTRENS WERE EXPAINED NEATLY