Topic : Hotel Reservation System for Weddings

Group no : MLB_08.01_01

Campus : Malabe

Submission Date:

We declare that this is our own work and this Assignment does not incorporate without acknowledgment any material previously submitted by anyone else in SLIIT or any other university/Institute. And we declare that each one of us equally contributed to the completion of this Assignment.

Registration No	**Name**	**Contact Number**
IT21164712	W.A.B.P Dhananjaya	077-3595823
IT21165702	V.U Jithma	077-4049074
IT21163340	H.R.T Peiris	076-6959956
IT21164576	M.A.D Sandeepani	071-0919058
IT21165566	S.K.S.M Sewwandi	076-3358772

Table of Contents

01. System Requirements

- The system should function 24/7/365 without the maintenance period.
- The Guest users can overview the system, to use the system they must register and create an account.
- When hotels, customers, and system staff register to the system they will get a unique user account.
- Every user should have an user accounts with unique username and password.
- The system should limit every single account to a single user.
- A hotel should make a payment and create an account.
- Customers and hotels should be able to edit their profiles.
- System Supports Customers Reserving hotels for their weddings according to their desire
- Reservations are made to the wedding halls in relevant hotels.
- Customers, as well as Guest users, can search hotels based on the hotel name, date, location, and Number of servings of their weddings.
- When a customer search for a hotel, the search result must contain filtered hotels according to the customer’s request.
- Customers should be able to cancel their bookings from their accounts within 48 hours.
- System Staff (Administrator, Moderator) should be able to accept reservations while only the administrator will be able to cancel reservations
- The system should calculate and view charges for the reservation.
- In the event of invalid input when filling the forms, the users should be sufficiently supported by the system to fill in the mandatory fields.
- Customers can book online and pay with a credit/debit card or online /bank transfer.
- The financial manager should verify payments.
- The system must send a booking confirmation email after successful payment.
- Customers should be able to make a list of their favorite hotels.
- Customers should be able to make an inquiry about their reservation.
- Customers can right reviews about the hotel and give ratings.
- Customers can check their booking status from their accounts.
- Hotels can publish their details (images, videos, etc.) on the website.
- System Staff (Administrator, Moderator) should verify the publications of the hotel before uploading.
- System Staff (Administrator, Moderator) should be able to publish the advertisement on the website.
- The hotel should update its calendar with confirm reservations.
- System Staff (Administrator, Moderator, Financial manager) should sign in to the system before engaging in any activity.
- The financial manager should provide reports on the reservation payments to the hotels.

02.Noun Verb Analysis

(Nouns are in red and Verbs are in Green)

- The system should function 24/7/365 without the maintenance period.
- The Guest users can overview the system, to use the system they must register and create an account.
- When hotels or customers or system staff register to the system they will get an unique user account.
- Every user should have an user accounts with unique username and password.
- The system should limit every single account to a single user.
- A hotel should make a payment and create an account.
- Customers and hotels should be able to edit their profiles.
- System Supports Customers Reserving hotels for their weddings according to their desire
- Reservations are made to the wedding halls in relevant hotels.
- Customers, as well as Guest users, can search hotels based on the hotel name, date, location, and Number of servings of their weddings.
- When a customer search for a hotel, the search result must contain filtered hotels according to the customer's request.
- Customers should be able to cancel their bookings from their accounts within 48 hours.
- System Staff (Administrator, Moderator) should be able to accept reservations while only the administrator will be able to cancel reservations.
- The system should calculate and view charges for the reservation.
- In the event of invalid input when filling the forms, the users should be sufficiently supported by the system to fill in the mandatory fields.
- Customers can book online and pay with a credit/debit card or online /bank transfer.
- The financial manager should verify payments.
- The system must send a booking confirmation email after successful payment.
- Customers should be able to make a list of their favorite hotels.
- Customers should be able to make an inquiry about their reservation.
- Customers can write reviews about the hotel and give ratings.
- Customers can check their booking status from their accounts.
- Hotels can publish their details (images, videos, etc.) on the website.
- System Staff (Administrator, Moderator) should verify the publications of the hotel before uploading.
- System Staff (Administrator, Moderator) should be able to publish the advertisement on the website.
- The hotel should update its calendar with confirm reservations.
- System Staff (Administrator, Moderator, Financial manager) should sign in to the system before engaging in any activity.
- The financial manager should provide reports on the reservation payments to the hotels.

2.1. Nouns

❖	Guest users	-	Class
❖	Account.	-	Redundant
❖	Hotel	-	Class
❖	Customers	-	Class
❖	System staff	-	Class
❖	User Accounts	-	Class
❖	User	-	Redundant
❖	Username	-	An attribute user account
❖	Password	-	An attribute user account
❖	Payment	-	Class
❖	Profiles	-	Redundant
❖	Weddings	-	Outside scope of system
❖	Wedding halls	-	Class
❖	Hotel name	-	Attribute of hotel
❖	Date	-	Attribute of resrvation
❖	Location	-	Attribute of resrvation
❖	Number of servings	-	Attribute of resrvation
❖	Search Result	-	An event or an operation
❖	Customer's request	-	An event or an operation
❖	Bookings	-	Redundant
❖	48 Hours	-	Outside scope of system
❖	Administrator	-	Class
❖	Moderator	-	Class
❖	Reservations	-	Class
❖	Charges	-	Attribute of reservation
❖	Invalid input	-	An event or an operation
❖	Forms	-	Outside scope of system

❖ Mandatory fields	-	Outside scope of system
❖ Credit/debit card	-	Outside scope of system
❖ Online /bank transfer.	-	Outside scope of system
❖ Financial manager	-	Class
❖ Booking confirmation	-	An event or an operation
❖ List of Favourite hotels	-	Attribute of class
❖ Inquiry	-	Class
❖ Reviews	-	Class
❖ Ratings	-	Class
❖ Booking status	-	Attribute of booking
❖ Details	-	Redundant
❖ Images	-	Attribute of hotel
❖ Videos	-	Attribute of hotel
❖ Publications	-	An event or an operation
❖ Advertisement	-	Class
❖ Calendar	-	Attribute of hotel
❖ Activity	-	An event or an operation
❖ Reports	-	Class

2.2. Identified Classes

1. Guest user
2. User Account
3. Hotel
4. Weddings hall
5. Customer
6. Reservation
7. Payment
8. Advertisement
9. Report
10. System staff
11. Administrator
12. Moderator
13. Financial manager
14. Inquiry
15. Reviews
16. Ratings

2.4 Methods

❖ Guest user	-	Overview the system Use the system Register to the system Create an account Search for filtered hotel
❖ User Account	-	Produce username Produce password
❖ Hotel	-	Make a payment Create an account Edit profiles Publish their details Update calendar Confirm reservations
❖ Customer	-	Edit profiles Make reservation Search for filtered hotel Cancel booking View charges for the reservation. Pay with a credit/debit card or online /bank transfer Receive confirmation email after successful payment Make an inquiry about their reservation Make a list of favorite hotels Write reviews Give ratings Check booking status
❖ Administrator	-	Accept reservation Cancel reservation Verify the publications of the hotel Publish the advertisement

- Moderator - Accept reservation
  Verify the publications of the hotel
  Publish the advertisement

- The financial manager - Verify payments
  Provide reports on the reservation payments

- System staff - Sign in to the system

03. CRC Cards

Class Name: Guest user	
Responsibility	**Collaborators**
Create an account	
Search for filtered hotel	
View Hotel	Hotel

Class Name: User Account	
Responsibility	**Collaborators**
System login/logout	Hotel, Customer, System staff
Change password	
Change username	

Class Name: Hotel	
Responsibility	**Collaborators**
Make a payment	Payment
Create an account	Administrator / Moderator
Edit profiles	
Publish their Details	
Update calendar	
Confirm Reservations	
Calculate ratings	Ratings

Class Name: Wedding hall	
Responsibility	**Collaborators**
Create wedding halls	Hotel
Update wedding halls details	Hotel
Delete wedding halls	Hotel

Class Name: Customer	
Responsibility	**Collaborators**
Edit profiles	
View Hotel	Hotel
Search the filtered hotels	
Make reservation	Reservation
Cancel booking	Reservation
Make payment	Payment
Check booking status	Reservation

Class Name: Reservation

Responsibility	Collaborators
Create Reservation	Customer
Update Reservation	Customer
Cancel Reservation	Customer, Administrator, Moderator
Calculate reservation fee	Financial Manager

Class Name: Payment

Responsibility	Collaborators
Make new payment	Customer, Hotel
Check payment details	
Confirm payments	

Class Name: Advertisement

Responsibility	Collaborators
Create Advertisement	Administrator, Moderator
Delete Advertisement	Administrator
Display advertisement status	

Class Name: Report	
Responsibility	**Collaborators**
Provide reports on the reservation payments	Financial manager
Store a list of payment history	Financial manager
Store a list of available dates of hotels	Hotel
Store a list of favorite hotels of customers	Customer
Store a list of reservation history	

Class Name: System staff	
Responsibility	**Collaborators**
Create Account	
Update Account	
Delete Account	

Class Name: Administrator	
Responsibility	**Collaborators**
Accept reservation	
Cancel reservation	
Verify the publications of hotels	
Publish the advertisements	

Class Name: Moderator	
Responsibility	**Collaborators**
Accept reservation	
Verify the publications of hotels	
Publish the advertisements	

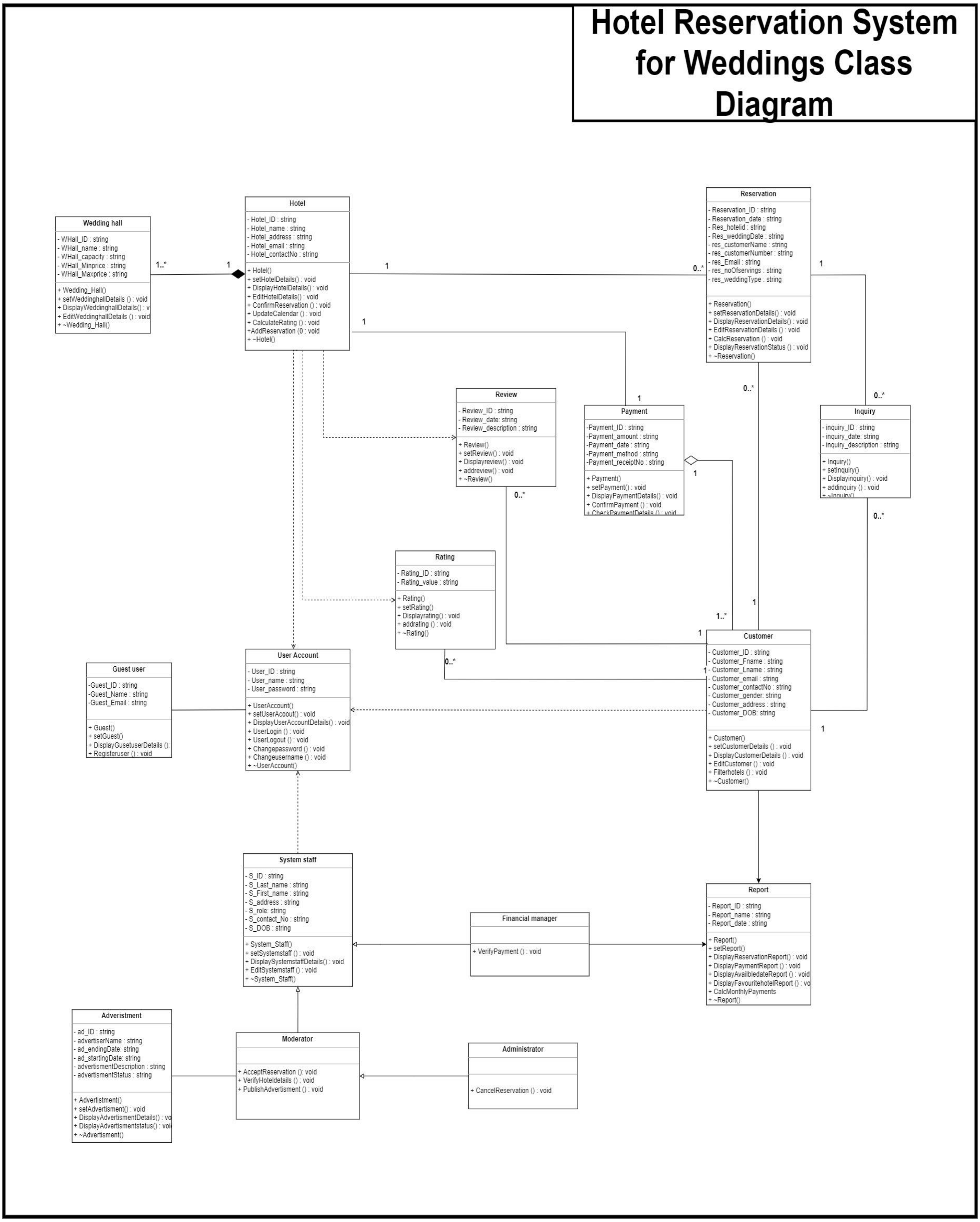
Class Name: Financial Manager	
Responsibility	**Collaborators**
Verify payments	
Sent confirmation email after successful payment	

Class Name: Inquiry	
Responsibility	**Collaborators**
Make an inquiry about their reservation	Customer
Display inquiries for a reservation	Reservation

Class Name: Reviews	
Responsibility	**Collaborators**
Write reviews	Customer
Display reviews	Hotel

Class Name: Ratings	
Responsibility	**Collaborators**
Give ratings	Customer
Display ratings	Hotel

04. Class Diagram (UML Notations)


Hotel Reservation System for Weddings Class Diagram
Wedding hall
- WHall_ID : string
- WHall_name : string
- WHall_capacity : string
- WHall_Minprice : string
- WHall_Maxprice : string
+ Wedding_Hall()
+ setWeddinghallDetails () : void
+ DisplayWeddinghallDetails() : v
+ EditWeddinghallDetails () : void
+ ~Wedding_Hall()
1..*
1
Hotel
- Hotel_ID : string
- Hotel_name : string
- Hotel_address : string
- Hotel_email : string
- Hotel_contactNo : string
+ Hotel()
+ setHotelDetails() : void
+ DisplayHotelDetails() : void
+ EditHotelDetails() : void
+ ConfirmReservation () : void
+ UpdateCalendar () : void
+ CalculateRating () : void
+AddReservation (0 : void
+ ~Hotel()
1
0..*
1
Reservation
- Reservation_ID : string
- Reservation_date : string
- Res_hotelid : string
- Res_weddingDate : string
- res_customerName : string
- res_customerNumber : string
- res_Email : string
- res_noOfservings : string
- res_weddingType : string
+ Reservation()
+ setReservationDetails() : void
+ DisplayReservationDetails() : void
+ EditReservationDetails () : void
+ CalcReservation () : void
+ DisplayReservationStatus () : void
+ ~Reservation()
1
0..*
0..*
1
Review
- Review_ID : string
- Review_date: string
- Review_description : string
+ Review()
+ setReview() : void
+ Displayreview() : void
+ addreview() : void
+ ~Review()
0..*
Payment
-Payment_ID : string
-Payment_amount : string
-Payment_date : string
-Payment_method : string
-Payment_receiptNo : string
+ Payment()
+ setPayment() : void
+ DisplayPaymentDetails() : void
+ ConfirmPayment () : void
+ CheckPaymentDetails () : void
1
Inquiry
- inquiry_ID : string
- inquiry_date: string
- inquiry_description : string
+ Inquiry()
+ setInquiry()
+ Displayinquiry() : void
+ addinquiry () : void
+ ~Inquiry()
0..*
Rating
- Rating_ID : string
- Rating_value : string
+ Rating()
+ setRating()
+ Displayrating() : void
+ addrating () : void
+ ~Rating()
0..*
1
1..*
1
1
1
Customer
- Customer_ID : string
- Customer_Fname : string
- Customer_Lname : string
- Customer_email : string
- Customer_contactNo : string
- Customer_gender: string
- Customer_address : string
- Customer_DOB: string
+ Customer()
+ setCustomerDetails () : void
+ DisplayCustomerDetails () : void
+ EditCustomer () : void
+ Filterhotels () : void
+ ~Customer()
Guest user
-Guest_ID : string
-Guest_Name : string
-Guest_Email : string
+ Guest()
+ setGuest()
+ DisplayGusetuserDetails ():
+ Registeruser () : void
User Account
- User_ID : string
- User_name : string
- User_password : string
+ UserAccount()
+ setUserAcoout() : void
+ DisplayUserAccountDetails() : void
+ UserLogin () : void
+ UserLogout () : void
+ Changepassword () : void
+ Changeusername () : void
+ ~UserAccount()
System staff
- S_ID : string
- S_Last_name : string
- S_First_name : string
- S_address : string
- S_role: string
- S_contact_No : string
- S_DOB : string
+ System_Staff()
+ setSystemstaff () : void
+ DisplaySystemstaffDetails() : void
+ EditSystemstaff () : void
+ ~System_Staff()
Financial manager
+ VerifyPayment () : void
Report
- Report_ID : string
- Report_name : string
- Report_date : string
+ Report()
+ setReport()
+ DisplayReservationReport() : void
+ DisplayPaymentReport () : void
+ DisplayAvailbledateReport () : void
+ DisplayFavouritehotelReport () : vo
+ CalcMonthlyPayments
+ ~Report()
Adveristment
- ad_ID : string
- advertiserName : string
- ad_endingDate: string
- ad_startingDate: string
- advertismentDescription : string
- advertismentStatus : string
+ Advertisment()
+ setAdvertisment() : void
+ DisplayAdvertismentDetails() : vo
+ DisplayAdvertismentstatus() : voi
+ ~Advertisment()
Moderator
+ AcceptReservation (): void
+ VerifyHoteldetails () : void
+ PublishAdvertisment () : void
Administrator
+ CancelReservation () : void

05. Coding for the classes

GuestUser.h

```cpp
#include <string>

using namespace std;

class Guest_user
{
private:
	string Guest_ID;
	string Guest_Name;
	string Guest_Email;
public:
	Guest_user(); //constructor
	void setGuest(string GID, string Gname, string GEmail);
	void DisplayGusetuserDetails();
	void Registeruser();
	void AddUserAccount(User_Account *U);

};
```

GuestUser.cpp

```cpp
#include "guestUser.h"
#include "userAccount.h"
#include <iostream>
#include <string>
#include <iomanip>

using namespace std;
Guest_user::Guest_user() {
	Guest_ID = "";
	Guest_Name = "";
	Guest_Email = "";
}
void Guest_user :: setGuest(string GID, string Gname, string GEmail) {
	Guest_ID = GID;
	Guest_Name = Gname;
	Guest_Email = GEmail;

};
```

```
void Guest_user::DisplayGusetuserDetails() {

};
void Guest_user::Registeruser() {

};
void Guest_user::AddUserAccount(User_Account* U) {


}
```

UserAccount.h

```
#include <string>

using namespace std;

class User_Account
{
private:
	string User_ID;
	string User_name;
	string User_password;
public:
	User_Account();
	void setUserAccount(string UID, string Uname, string Upassword);
	void UserAccount();
	void DisplayUserAccountDetails();
	void UserLogin();
	void UserLogout();
	void Changepassword();
	void Changeusername();


};
```

UserAccount.cpp

```
#include "userAccount.h"
#include <iostream>
#include <string>
#include <iomanip>

using namespace std;

User_Account::User_Account() {
	User_ID = "";
	User_name = "";
	User_password = "";
}

void User_Account::setUserAccount(string UID, string Uname, string Upassword) {
	User_ID = UID;
	User_name = Uname;
	User_password = Upassword;

};

void DisplayUserAccountDetails() {

};
void UserLogin() {

};
void UserLogout() {

};
void Changepassword() {

};
void Changeusername() {


};
```

Hotel.h

```
#include <string>
#include "WeddingHall.h"
#include "Reservation.h"
#include "UserAccount.h"
#include "Review.h"
#include "Payment.h"
#include "Rating.h"
using namespace std;
class hotel //Hotel class
{
private :  // Attributies
      string Hotel_ID;
      string Hotel_name;
      string Hotel_address;
      string Hotel_email;
      string Hotel_contactNo;
      Wedding_Hall* weddingHall[SIZE];
      Reservation * Res[SIZE];
      Payment * pay;
public: // Methods

      hotel();
      void SetHotelDetails(string HID, string Hname, string Haddress,
string Hemail, string Srole, string HcontactNo, Wedding_Hall*
weddingHall);
      void DisplayHotelDetails();
      void DelHotelDetails();
      void EditHotelDetails();
      void ConfirmReservation();
      void UpdateCalendar();
      void CalculateRating();
      void DisplayWedding_Halls();
      void Addhotel(int acc, User_Account *U);
      void Addhotel(int rev, review *R);
      void Addhotel(int rate, ratings *Rt);
      ~hotel();
};
```

Hotel.cpp

```
#include "Hotel.h"
#include "WeddingHall.h"
#include "Reservation.h"
#include "UserAccount.h"
#include "Review.h"
#include "Payment.h"
#include "Rating.h"
#include <iostream>
#include <string>
#include <iomanip>

using namespace std;
```

```
hotel::hotel()// Default constructor
{
        Hotel_ID = "";
        Hotel_name = "";
        Hotel_address = "";
        Hotel_email = "";
        Hotel_contactNo = "";
        weddingHall[0] = new Wedding_Hall();
        weddingHall[1] = new Wedding_Hall();
       hotel -> AddReservation();
};
void hotel::SetHotelDetails (string HID, string Hname, string Haddress,
string Hemail, string Srole, string HcontactNo,int w1, int w2)// Default
constructor
{
      Hotel_ID = HID;
      Hotel_name = Hname;
      Hotel_address = Haddress;
      Hotel_email = Hemail;
      Hotel_contactNo = HcontactNo;
      weddingHall[0] = new Wedding_Hall(w1);
      weddingHall[1] = new Wedding_Hall(w2);

};
void hotel::DisplayHotelDetails()// Method{
};

void hotel::DelHotelDetails()// Method
{

};
void hotel::EditHotelDetails()// Method
{

};
void hotel::ConfirmReservation()// Method
{
};

void hotel::UpdateCalendar()// Method
{

};

void hotel::CalculateRating()// Method
{

};
void hotel::DisplayWedding_Halls() {
      for (int w = 0; w < 2; w++)
            weddingHall(w)->Display();
}
void hotel::AddReservation(Reservation* R) {

}
```

```
void hotel::AddPayment(Payment* P) {

}
void Addhotel(int acc, User_Account* U) {

}
void Addhotel(int rev, review* R) {

}
void Addhotel(int rate, ratings* Rt) {

}
~hotel() {
        cout << "Hotel is closing " << endl;
        //when the whole object is removed
        for (int w = 0; w < 2; w++)
                delete weddingHall[w];
        cout << "Everything is deleted" << endl;

}
```

WeddingHall.h

```
#include <string>
using namespace std;

class Wedding_Hall
{
private:
        string WHall_ID;
        string WHall_name;
        string WHall_capacity;
        string WHall_Minprice;
        string WHall_Maxprice;
public:
        Wedding_Hall();
        void setWedding_Hall(string WID, string Wname, string Wcapacity, string
        Wminprice, string Wmaxprice);
        void DisplayWedding_HallDetails();
        void EditWedding_Hall();
        ~Wedding_Hall();
};
```

WeddingHall.cpp

```
#include "WeddingHall.h"
#include <iostream>
#include <cstring>
#include <iomanip>

using namespace std;
```

```
Wedding_Hall::Wedding_Hall() {

	WHall_ID = "";
	WHall_name = "";
	WHall_capacity = "";
	WHall_Minprice = "";
	WHall_Maxprice = "";
};
void Wedding_Hall::setWedding_Hall(string WID, string Wname, string
Wcapacity, string Wminprice, string Wmaxprice) {
	WHall_ID = WID;
	WHall_name = Wname;
	WHall_capacity = Wcapacity;
	WHall_Minprice = Wminprice;
	WHall_Maxprice = Wmaxprice;
};
void Wedding_Hall::DisplayWedding_HallDetails() {
}
void Wedding_Hall::EditWedding_Hall() {

}
```

Customer.h

```
//customer.h
#include<iostream>
#include<string>
#include "reporth.h"
#include "Rating.h"
#include "Review.h"
using namespace std;
class customer
{
private:
    string Customer_ID;
    string Customer_Fname;
    string Customer_Lname;
    string Customer_Email;
    string Customer_ContactNo;
    string Customer_Gender;
    string Customer_Address;
    string Customer_DOB;
    Report  *RPT[2];
    ratings *rating[2];
    review *rview[2];

public:
    customer();//constructor
    void setCustomerDetails(string CID, string C_Fname, string C_Lname,
string C_Email, string C_contactNo, string C_gender, string C_address,
string C_DOB);
void DisplayCustomerDetails();
    void EditCustomer();
    void Filterhotels();
    ~customer();//destructor

};
```

Customer.cpp

```
#include "customerh.h"
#include "reporth.h"
#include <iostream>
#include <string>
#include <iomanip>

using namespace std;

customer::customer() {
     Customer_ID = "";
     Customer_Fname = "";
     Customer_Lname = "";
     Customer_Email = "";
     Customer_ContactNo = "";
     Customer_Gender = "";
     Customer_Address = "";
     Customer_DOB = "";
     Report *RPT;
};

void customer::setCustomerDetails(string CID, string C_Fname, string 
C_Lname, string  C_Email, string C_contactNo, string C_gender, string 
C_address, string C_DOB) {

     Customer_ID = CID;
     Customer_Fname = C_Fname;
     Customer_Lname = C_Lname;
     Customer_Email = C_Email;
     Customer_ContactNo = C_contactNo;
     Customer_Gender = C_gender;
     Customer_Address = C_address;
     Customer_DOB = C_DOB;
     Report = RPT;
};

void customer::DisplayCustomerDetails() {

};
void customer::EditCustomer() {

};
void customer::Filterhotels() {

};
```

Reservation.h

```
//Reservation class
#include <string>
#include "inquaryh.h"
#include "customerh.h"
#include "hotel.h"

using namespace std;

class Reservation
{
private:
	string Reservation_ID;
	string Reservation_date;
	string Res_hotelid;
	string Res_weddingDate;
	string Res_customerName;
	string Res_customerNumber;
	string Res_Email;
	string Res_noOfservings;
	string Res_weddingType;
	Inquiry*inq[SIZE];
	customer*cus;
	hotel*htl;

public:
	Reservation();//Default constructor
	Reservation(string
Reservation_ID,hotel*htel,customer*cust);//overload constructor
	void setReservationDetails(string R_ID, string R_date, string
R_hotelID, string R_weddingDate, string R_cname, string R_c_no, string
R_email, string R_Sno, string R_wType);
	void addInquiry(Inquiry*I);
	void DisplayReservationDetails();
	void EditReservationDetails();
	void CalcReservation();
	void DisplayReservationStatus();
	~Reservation();//destructor
};
```

Resrvation.cpp

```
#include "reservationh.h"
#include "inquaryh.h"
#include "customerh.h"
#include "hotel.h"
#include <iostream>
#include <string>
#include <iomanip>

using namespace std;
```

```
Reservation::Reservation()
{
    Reservation_ID = "";
    Reservation_date = "";
    Res_hotelid = "";
    Res_weddingDate = "";
    Res_customerName = "";
    Res_customerNumber = "";
    Res_Email = "";
    Res_noOfservings = "";
    Res_weddingType = "";
};
void Reservation::setReservationDetails(string R_ID, string R_date, string
R_hotelID, string R_weddingDate, string R_cname, string R_c_no, string
R_email, string R_Sno, string R_wType)
{
    Reservation_ID = R_ID;
    Reservation_date = R_date;
    Res_hotelid = R_hotelID;
    Res_weddingDate = R_weddingDate;
    Res_customerName = R_cname;
    Res_customerNumber = R_c_no;
    Res_Email = R_email;
    Res_noOfservings = R_Sno;
    Res_weddingType = R_wType;
};
Reservation::Reservation(string Reservation_ID,hotel*htel,customer*cust)
{

};
void Reservation::addInquiry(Inquiry*I)
{

};
void Reservation::DisplayReservationDetails()
{

};
void Reservation::EditReservationDetails()
{

};
void Reservation::CalcReservation()
{

};
void Reservation::DisplayReservationStatus() {
};

Reservation::~Reservation()
{
     for (int i = 0; i < SIZE; i++)
     delete Reservation[i];
     cout << "Everything is deleted" << endl;
};
```

Payment.h

```
#include <string>
#include "customerh.h"
#include "hotel.h"

using namespace std;

class Payment
{
private:
    string Payment_ID;
    string Payment_amount;
    string Payment_date;
    string Payment_method;
    string Payment_receiptNo;
    customer* cus[SIZE];
    hotel *htl;



public:
    Payment();//Default constructor
    void setPayment(string PayID, string Pay_amount, string Pay_date, string 
Pay_method, string Pay_reciptNo);
    void addhotel(hotel*H);
    void DisplayPaymentDetails();
    void ConfirmPayment();
    void CheckPaymentDetails();
    void addcustomer(customer *c1, customer *c2);
    ~Payment();//destructor
};
```

Payment.cpp

```
#include "paymenth.h"
#include <iostream>
#include "customerh.h"
#include "hotel.h"
#include <cstring>
#include <iomanip>

using namespace std;
Payment::Payment()
{
    Payment_ID = "";
    Payment_amount = "";
    Payment_date = "";
    Payment_method = "";
    Payment_receiptNo = "";
};
```

```
void Payment::setPayment(string PayID, string Pay_amount, string Pay_date,
string Pay_method, string Pay_reciptNo)
{
    Payment_ID = PayID;
    Payment_amount = Pay_amount;
    Payment_date = Pay_date;
    Payment_method = Pay_method;
    Payment_receiptNo = Pay_reciptNo;

};
void Payment::addhotel(hotel*H)
{
};

void Payment::DisplayPaymentDetails()
{
      for(int i=0; i<SIZE;i++)
      {
            cus[i]->DisplayCustomerDetails();}
};

void Payment::ConfirmPayment()
{
};

void Payment::CheckPaymentDetails()
{
};

void addcustomer(customer* c1, customer* c2)
{
    cus[0] = c1;
    cus[1] = c2;
};

Payment::~Payment()
{
   cout << "Everything is deleted" << endl;
};
```

Advertisment.h

```
#include <string>

using namespace std;

class Advertisment
{
private:
      string ad_ID;
      string advertiserName;
```

```
     string ad_endingDate;
     string ad_startingDate;
     string adDescription;
     string adStatus;

public:
     Advertisment();//constructor
     void setAdvertisment(string AdID, string Adname, string Ad_endDate,
string Ad_startDate, string AdDes, string Ad_status);
     void DisplayAdvertismentDetails();
     void DisplayAdvertismentstatus();
     ~Advertisment(); //destructor

};
```

Advertisment.cpp

```
#include "Advertisment.h"
#include <iostream>
#include <string>
#include <iomanip>

using namespace std;

Advertisment::Advertisment() {
      ad_ID = "";
      advertiserName = "";
      ad_endingDate = "";
      ad_startingDate = "";
      adDescription = "";
      adStatus = "";

};

void Advertisment::setAdvertisment(string AdID, string Adname, string
Ad_endDate, string Ad_startDate, string AdDes, string Ad_status){

     ad_ID = AdID;
     advertiserName = Adname;
     ad_endingDate = Ad_endDate;
     ad_startingDate = Ad_startDate;
     adDescription = AdDes;
     adStatus = Ad_status;

};

void Advertisment::DisplayAdvertismentDetails() {

};

void Advertisment::DisplayAdvertismentstatus() {

}
```

Report.h

```
#include <string>

using namespace std;

class Report
{
private:
	string Report_ID;
	string Report_name;
	string Report_date;

public:
	Report();
	void setReport(string RID, string Rname, string Rdate);
	void DisplayReservationReport();
	void DisplayPaymentReport();
	void DisplayAvailbledateReport();
	void DisplayFavouritehotelReport();
	void CalcMonthlyPayments();
	~Report ();
};
```

Report.cpp

```
#include "Report.h"
#include <iostream>
#include <string>
#include <iomanip>

using namespace std;

Report::Report() {
    Report_ID = "";
    Report_name = "";
    Report_date = "";

}

void Report::setReport(string RID, string Rname, string Rdate) {
    Report_ID = RID;
    Report_name = Rname;
    Report_date = Rdate;
};

void Report::DisplayReservationReport() {

};
void Report::DisplayPaymentReport() {

};
```

```
void Report::DisplayAvailbledateReport() {

};
void Report::DisplayFavouritehotelReport() {

};
void Report::CalcMonthlyPayments() {


};
```

Systemstaff.h

```
#include <string>
using namespace std;

class System_Staff
{
private:
	string S_ID;
	string S_First_name;
	string S_Last_name;
	string S_address;
	string S_role;
	string S_contact_No;
	string S_DOB;
public:
	System_Staff();//constructor
	void setSystemstaff(string SID, string SFirst_name, string SLast_name,
string Saddress, string Srole, string Scontact_No, string SDOB);
	void DisplaySystemstaffDetails();
	void EditSystemstaff();
	~System_Staff();//destructor
};
```

Systemstaff.cpp

```
#include "Systemstaff.h"
#include <iostream>
#include <string>
#include <iomanip>

using namespace std;

System_Staff::System_Staff() {
    S_ID = "";
    S_First_name = "";
    S_Last_name = "";
    S_address = "";
    S_role = "";
    S_contact_No = "";
    S_DOB = "";

};

void System_Staff::setSystemstaff(string SID, string SFirst_name, string
SLast_name, string Saddress, string Srole, string Scontact_No, string SDOB) {
    S_ID = SID;
    S_First_name = SFirst_name;
    S_Last_name = SLast_name;
    S_address = Saddress;
    S_role = Srole;
    S_contact_No = Scontact_No;
    S_DOB = SDOB;

};

void System_Staff::DisplaySystemstaffDetails() {

};

void System_Staff::EditSystemstaff() {

};
```

Administrator.h

```
#include "Moderator.h"

class Administrator : public Moderator
{
public:
	void PublishAdvertistment();

};
```

Administrator.cpp

```
#include "Adminstratorh.h"

void Administrator::PublishAdvertistment() {

};
```

Moderator.h

```
#include "Systemstaff.h"

class Moderator : public System_Staff
{
public:
	void AcceptReservation();
	void CancelReservation();
	void VerifyReservation();
};
```

Moderator.cpp

```
#include "Moderator.h"

void Moderator::AcceptReservation() {

};

void Moderator::CancelReservation() {

};

void Moderator::VerifyReservation() {

};
```

FinancialManager.h

```
#include "systemstaff.h"

class Financial_Manager : public System_Staff
{
public:
	void VerifyPayment();
};
```

FinancialManager.cpp

```
#include "FinancialManager.h"

void Financial_Manager::VerifyPayment() {


};
```

Inquiry.h

```
#include <cstring>
#include "reservationh.h"
#include "customerh.h"
using namespace std;

class Inquiry
{
private:
	string inquiry_ID;
	string inquiry_date;
	string inquiry_description;
	Reservation*Res;
	customer*cus;

public:
	Inquiry();//Default constructor
	Inquiry(string inquiry_ID,customer*cust,Reservation*Resrv);//overload
constructor
	void setInquiry(string i_ID, string i_date, string i_description);
	void Displayinquiry();
	void addinquiry();
	~Inquiry();//destructor
};
```

Inquiry.cpp

```
#include "inquaryh.h"
#include "reservationh.h"
#include "customerh.h"
#include <iostream>
#include <cstring>
#include <iomanip>

using namespace std;

Inquiry::Inquiry() {
	inquiry_ID = "";
	inquiry_date = "";
	inquiry_description = "";

};
void Inquiry::setInquiry(string i_ID, string i_date, string
i_description) {
	inquiry_ID = i_ID;
	inquiry_date = i_date;
	inquiry_description = i_description;
};

Inquiry::Inquiry(string inquiry_ID,customer*cust,Reservation*Resrv)
{
};

void Inquiry::Displayinquiry() {

};
void Inquiry::addinquiry() {

};
Inquiry::~Inquiry()
{
	 for (int i = 0; i < SIZE; i++)
	     delete Inquiry[i];
	cout << "Everything is deleted" << endl;
};
```

Review.h

```
#include<string>
using namespace std;

class review {

private:
	string Review_ID;
	string Review_date;
	string Review_description;
private:
	review();//constructor
	void setReview(string R_ID, string R_date, string R_descriptionn);
	void Displayreview();
	void addreview();
	~review();//destructor

};
```

Review.cpp

```
#include "Review.h"
#include <iostream>
#include <string>
#include <iomanip>

using namespace std;

review::review() {
	Review_ID = "";
	Review_date = "";
	Review_description = "";

};
void review::setReview(string R_ID, string R_date, string R_description) {

	Review_ID = R_ID;
	Review_date = R_date;
	Review_description = R_description;

};
void review::Displayreview() {

};
void review::addreview() {

};
```

Rating.h

```
//Rating.h
#include<string>
using namespace std;

class ratings {

private:
	string Rating_ID;
	string Rating_value;

public:
	ratings();//constructor

	void setRating(string R_ID, string R_value);
	void Displayrating();
	void addrating();
	~ratings();//destuctor


};
```

Rating.cpp

```
#include "Rating.h"
#include <iostream>
#include <string>
#include <iomanip>

using namespace std;

ratings::ratings() {

    Rating_ID = "";
    Rating_value = "";
};

void ratings::setRating(string R_ID, string R_value) {

    Rating_ID = R_ID;
    Rating_value = R_value;
};
void ratings::Displayrating() {

};
void ratings::addrating() {

};
```

5.3 Main Program

```
#include "GuestUser.h" // Guest User Header file
#include "UserAccount.h" //User account Header file
#include "Hotel.h" // hotel Header file
#include "WeddingHall.h" // wedding Hall Header file
#include "Customer.h" // customer Header file
#include "Reservation.h" // reservation Header file
#include "Payment.h" // payment Header file
#include "Advertisment.h" // advertisment Header file
#include "Report.h" //report Header file
#include "Systemstaff.h" // systemStaff Header file
#include "Administrator.h" //administrator Header file
#include "Moderator.h" // Moderator Header file
#include "FinancialManager.h" // Financial Manager Header file
#include "Inquiry.h" // Inquiry Header file
#include "Review.h" // Review Header file
#include "Rating.h" // Rating Header file
#include<iostream> // Standard input - output Header file
#include<string> // string header file
#include<iomanip> // input output header file
#define SIZE

using namespace std; // Prefix for std::

int main() // Main Function
{
     Report *RPT1; // customer class and Report class uni directional
relationship
     RPT1 = new Report();
     RPT1->generateReport(); // Display Generate Report

     Customer *cus1; // customer class and Report class uni directional
relationship
     cus1 = new Customer();
     cus1->displayCustomerDetails(); // Display customer details

     FinancialManager *FM1; // FinancialManager class and Report class uni
directional relationship
     FM1 = new FinancialManager();
     FM1->displayFinancialManagerDetails(); // DisplayFinancialManager details

     Payment *P1; // Payment Class and customer class Aggregation relationship
     P1 = new Payment();
     P1->displayPaymentDetails(); // Display Payment details
```

```
    Hotel *H1; // Hotel class and weddingHall class composition relationship
    H1 = new Hotel();
    H1->displayHotelDetails(); // Display Hotel details

    Customer *cus2 = Customer(); // Association Relationship between Customer
class and reservation class
    Reservation *R1 = new Reservation(); // Association Relationship Customer
class and reservation class

    Customer *cus3 = Customer(); // Association Relationship between Customer
class and Inquiry class
    Inquiry *I1 = new Inquiry(); // Association Relationship Customer class
and Inquiry class

    Customer *cus4 = Customer(); // Association Relationship between Customer
class and Review class
    Review  *Re1 = new Review (); // Association Relationship Customer class
and Review  class


    Customer *cus5 = Customer(); // Association Relationship between Customer
class and Rating class
    Rating  *Ra1 = new Rating (); // Association Relationship Customer class
and Rating  class


    Reservation *R2 = new Reservation(); // Association Relationship between
Reservation class and Inquiry class
    Inquiry *I2 = new Inquiry(); // Association Relationship Reservation
class and Inquiry class


    Reservation *R3 = new Reservation(); // Association Relationship between
Reservation class and Hotel class
    Hotel *H2 = new Hotel(); // Association Relationship Reservation class
and Hotel class


    Payment *P2 = new Payment(); // Association Relationship between Payment
class and Hotel class
    Hotel *H3 = new Hotel(); // Association Relationship Payment class and
Hotel class


    System_Staff *Ad1 = new Administrator(); // Inheritance relationship
administrator class
    System_Staff *M1 = new Moderator(); // Inheritance relationship Moderator
class
    System_Staff *FM2 = new FinancialManager(); // Inheritance relationship
FinancialManager class
```

```
Moderator *M2 = new Moderator(); // Association Relationship between
Moderatorclass and Advertisment class

	Advertisment *Ad2 = new Advertisment(); // Association Relationship
Moderator class and Advertismentclass


	UserAccount *U1 = new UserAccount(); // Association Relationship between
UserAccount class and GuestUser class
	GuestUser *G1 = new GuestUser(); // Association Relationship
UserAccountclass and GuestUser class


	Hotel *H4 = new Hotel(); // Dependency Relationship Hotel class and user
account class
	UserAccount *U2 = new UserAccount(); // Dependency Relationship Hotel
class and user account class


	Hotel *H5 = new Hotel(); // Dependency Relationship Hotel class and
review class
	Review  *Re2 = new Review (); // Dependency Relationship Hotel class and
review class


	Hotel *H6 = new Hotel(); // Dependency Relationship Hotel class and
rating class
	Rating  *Ra2 = new Rating ();  // Dependency Relationship Hotel class and
rating class


	UserAccount *U3 = new UserAccount(); // Dependency Relationship user
account class and system staff class
	System_Staff *S1 = new System_Staff (); // Dependency Relationship user
account class and system staff class


	UserAccount *U4 = new UserAccount(); // Dependency Relationship user
account class and customer class
	Customer *cus6 = Customer(); // Dependency Relationship user account
class and customer class


	delete RPT1; // Delete rept 1
	delete cus1;//delete cus1;
	delete FM1;//delete FM1;
	delete P1; //delete P1;
	delete H1; // delete H1;
	delete cus2; //delete cus2;
	delete R1; //delete R1;
	delete cus3; //delete cus3;
```

```
    delete I1; // delete I1;
    delete cus4; //delete cus4;
    delete Re1; //delete Re1;
    delete Ra1; //delete Ra1;
    delete R2; //delete R2;
    delete I2; //delete I2;
    delete R3; //delete R3;
    delete H2; //delete H2;
    delete P2; //delete P2;
    delete H3; //delete H3;
    delete Ad1; //delete Ad1;
    delete M1; //delete M1;
    delete FM2;//delete FM2;
    delete U1; //delete U1;
    delete G1; //delete G1;
    delete H4; //delete H4;
    delete U2, //delete U2,
    delete H5; //delete H5;
    delete Re2; //delete Re2;
    delete H6; //delete H6;
    delete Ra2; //delete Ra2;
    delete U3; //delete U3;
    delete S1; //delete S1;
    delete U4; //delete U4;
    delete cus6; //delete cus6;

    return 0; // End of Main function

}
```

Individual Contributions

	Student ID	Student Name	Individual Contribution
1	IT21164812	W.A.B.P Dhananjaya	• Report.h and Report.cpp File • GuestUser.h and GuestUser.cpp File • UserAccount.h File and UserAccount.cpp File
2	IT21165702	V.U.Jithma	• Systemstaff.h and Systemstaff.cpp File • FinancialManager.h and FinancialManager.cpp File • Administrator.h Administrator.cpp File • Moderator.h and Moderator.cpp File
3	IT21163340	H.R.T Peiris	• Hotel.h and Hotel.cpp File • WeddingHall.h and WeddingHall.cpp File • Advertisment.h and Advertisment.cpp File • Main.cpp File
4	IT21164576	M.A.D.Sandeepani	• Payment.h and Payment.cpp File • Reservation.h and Reservation.cpp File • Inquiry.h and Inquiry.cpp File
5	IT21165566	S.K.S.M Sewwandi	• Customer.h and Customer.cpp File • Review.h and Review.cpp File • Ratings.h and Ratings.cpp File