Topic : Hotel Reservation System

Group no : MLB_07.02_12

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
IT21354588	R.M.M.V Rathnayake	+94 78 991 0357
IT21484582	Nanayakkara H.	+94 71 277 2650
IT21356940	Ranasinghe R.T.A	+94 77 070 0584
IT21263026	Shukri H.M	+94 77 323 5236
IT21352058	M.S.M Abeyrathne	+94 76 848 3294

1) System Requirements

1) A Guest user can search for details of the hotel and services provided.
2) Guest users have to get registered in the system to make reservations or any type of service.
3) The user can contact the receptionist to get further information.
4) The system has several types of user accounts, according to the type of services they receive and provide.
5) The system keeps track records of all users.
6) Effective user interface for users to easily navigate the reservation process.
7) Registered users have reliable access anytime they wanted.
8) The system will be shown how the system works and will be provided with links to access before registration.
9) In the contact us method, users can see the contact details of hotel receptionists and the hotel location.
10) Users can view the facilities that we supply to give a better service for them.
11) Users can reserve a banquet hall, according to their needs.
12) When a registered user makes a reservation, they can customize the packages according to their requirements. (Terms and conditions applied).
13) A registered user can create a favourite package and save it for easy reuse.
14) A registered user can make a reservation or cancel a reservation (before 7 days of the reservation), choose packages, and make payments.
15) Registered users can receive refunds when cancelled their reservations. (terms and conditions applied)
16) When the reservation is confirmed, the event manager will contact the registered user.
17) Decoration agencies, Entertainment companies can also register their companies in the system and expand their reach.
18) Decoration agents, Entertainment agents can add new services to the system and remove services they provide.
19) All users can choose their payment options as they preferred.
20) The user should be able to manage online payments and cash payments.
21) Transaction methods should be secured to fulfil user security requirements.
22) System admin can block any user or agents who harm the reputation and goodwill of the system.
23) The system should enable web services such as social media sharing services and blogs for users to share experiences and interact with each other.
24) System admin can generate reports (economy, usage, popularity) according to higher management's criteria.
25) Users can give feedback regarding the quality of service.

2) Noun and Verbs Analysis

Noun	
User, the Guest user	Redundant
Hotel, The system, Higher management, Decoration agencies, Entertainment companies, Decoration agencies, Companies, Entertainment agents, social media	Out of scope
Search details	An event or operation
Criteria, terms, conditions	Meta-language
Types of services, Quality of services	Meta-language
Economy, usage, popularity (Report)	Attributes
Guest user	Class
Registered user	Class
Receptionist	Class
Banquet hall	Class
Event manager	Class
System admin	Class
Payment	Class
Report	Class
Feedback	Class

Verbs	
Search, Get, See, View, Reserve, Manage	Guest User
Create, Save, Reuse, Cancel, Choose, Receive, Access, Give	Registered Users
Contact	Event manager
Block, Generate	System admin

Identified classes

- Guest user
- Registered user
- Receptionist
- Banquet hall
- Event manager
- System admin
- Payment
- Report
- Feedback

3) CRC Cards for the System

Guest user	
Responsibility	**Collaborators**
Search hotel	
Payment	
Register	

Registered user	
Responsibility	**Collaborators**
Reliable access	
View facilities	
Reserve and cancel reserve	Banquet hall

Receptionist	
Responsibility	**Collaborators**
Contact us	Registered user

Banquet hall	
Responsibility	**Collaborators**
Store banquet hall details	

Event manager	
Responsibility	**Collaborators**
Reservation, customized packages	Registered user

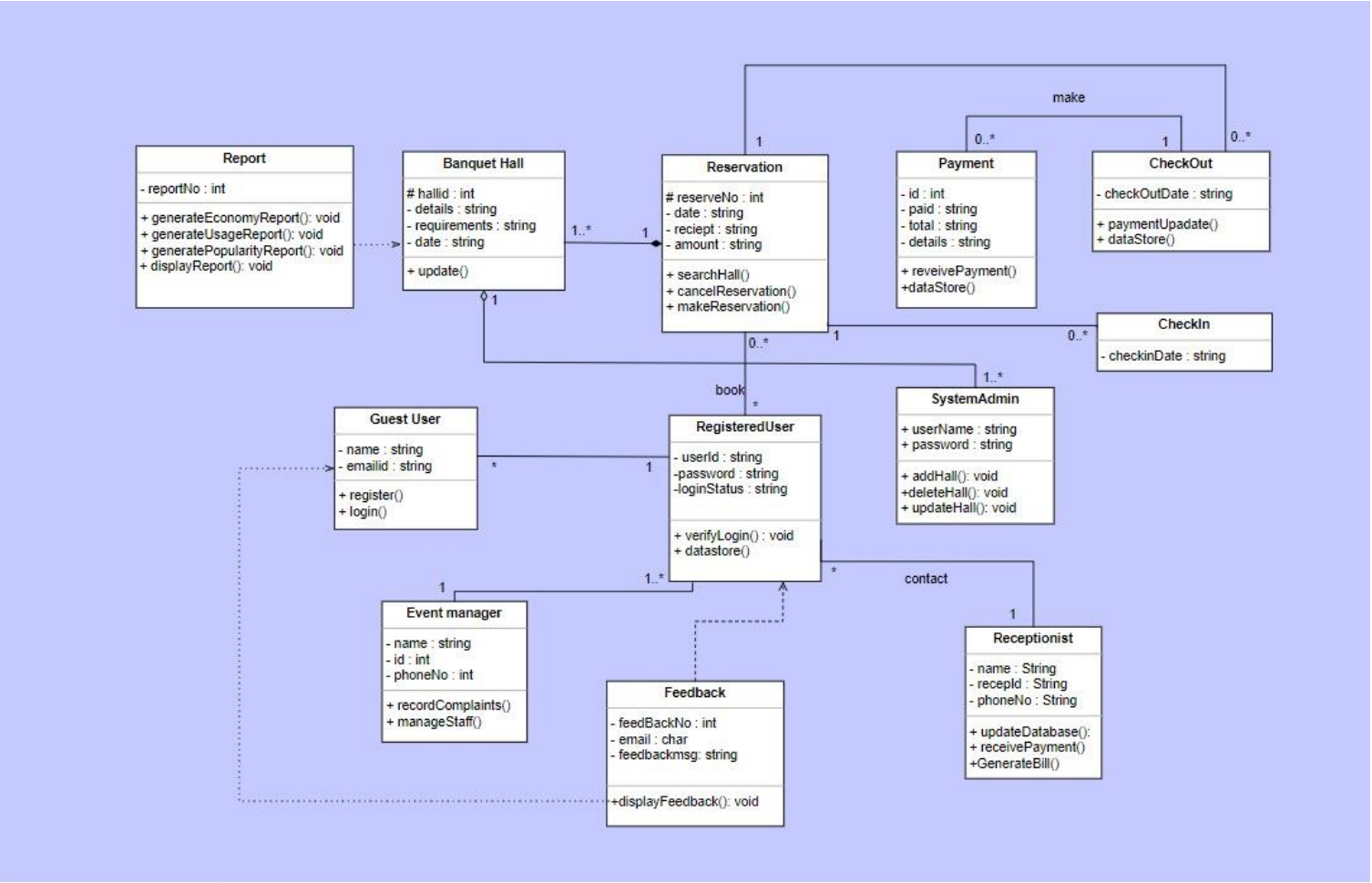
System admin	
Responsibility	**Collaborators**
View and update user feedbacks	Feedback
Edit and update banquet hall details	Banquet hall
update the availability status of booking halls	

Payment	
Responsibility	**Collaborators**
Add payment	
Check payment details	System Admin
Confirm payment	

Report	
Responsibility	**Collaborators**
Generate payment report	Payment

Feedback	
Responsibility	**Collaborators**
Store feedback details	Registered user

4) Class Diagram



5) Code

```
#include <cstring>

#include <iostream>

using namespace std;

#define SIZE 2

#define SIZE2 2

#define SIZE2 3

#define SIZE3 4

#define SIZE4 2
```

RegisteredUser

```
class RegisteredUser

{

private:

        int userid;

        char password[50];

        char loginStatus[20];

        EventManager* em;

        GuestUser* gu[SIZE];

        Reservation* re[SIZE2];

        Receptionist* rc;

public:

        RegisteredUser();

        void verifyLogin();

        void dataSotre();

        ~RegisteredUser();

};
```

```
RegisteredUser::RegisteredUser()
{
}
void RegisteredUser::verifyLogin()
{
}
void RegisteredUser::dataSotre()
{
}
RegisteredUser::~RegisteredUser()
{
}
```

Reservation

```
class Reservation
{
protected:
        int reserveNo;
private:
        char date[20];
        char reciept[20];
        char amount[20];

        CheckIn* chk[SIZE];
        Checkout* chkout[SIZE2];
        BanquetHall* bhall[SIZE3];
        RegisteredUser* ru[SIZE4];
```

```
public:
        void searchHall();
        void cancelReservation();
        void makeReservation();
        Reservation();
        ~Reservation();
};
void Reservation::searchHall()
{
}
void Reservation::cancelReservation()
{
}
void Reservation::makeReservation()
{
}
Reservation::Reservation()
{
}
Reservation::~Reservation()
{
}
```

GuestUser

```
class GuestUser
{
private:
        char name[50];
        char emailid[20];

        RegisteredUser* ru;
public:
        GuestUser();
        void uregister();
        void login();
        ~GuestUser();
};
GuestUser::GuestUser()
{
}
void GuestUser::uregister()
{
}
void GuestUser::login()
{
}
GuestUser::~GuestUser()
{
}
```

Receptionist

```
class Receptionist
{
private:
        char name[30];
        int recpId;
        char phoneNo[10];

        RegisteredUser* ru[SIZE];
public:
        Receptionist();
        void updateDatabse();
        void receivePayment();
        void generateBill();
        ~Receptionist();
};
Receptionist::Receptionist()
{
}
void Receptionist::updateDatabse()
{
}
void Receptionist::receivePayment()
{
}
void Receptionist::generateBill()
```

```
{

}

Receptionist::~Receptionist()

{

}
```

Payment

```
class Payment

{

private:

        int id;

        char paid[20];

        char total[10];

        char details[20];


        Checkout* ck;


public:

        Payment();

        void reservePayment();

        void dataStore();

        ~Payment();

};

Payment::Payment()

{

}

void Payment::reservePayment()
```

```
{

}

void Payment::dataStore()

{

}

Payment::~Payment()

{

}
```

BanquetHall

```
class BanquetHall

{

protected:

        int hallid;

private:

        char details[20];

        char requirments[20];

        char date[20];


        SystemAdmin* sys[2];

public:

        void update();

        BanquetHall()

        ~BanquetHall();

};

BanquetHall::BanquetHall()

{
```

```
}
void BanquetHall::update()
{
}
BanquetHall::~BanquetHall()
{
}
```

CheckIn

```
class CheckIn
{
private:
	char checkinDate[20];
	Reservation* res;
public:
	CheckIn();
	~CheckIn();
};
CheckIn::CheckIn()
{
}
CheckIn::~CheckIn()
{
}
```

Checkout

```
class Checkout
{
private:
	char checkoutDate[30];
	Reservation* r;
	Payment* pay[SIZE];
public:
	Checkout();
	void paymentUpdate();
	void dataStore();
	~Checkout();
};
Checkout::Checkout()
{
}
void Checkout::paymentUpdate()
{
}
void Checkout::dataStore()
{
}
Checkout::~Checkout()
{
}
```

EventManager

```
class EventManager
{
private:
	char name[20];
	int id;
	int phoneNo;

	RegisteredUser* ru[SIZE];
public:
	EventManager();
	void recordComplaints();
	void manageStaff();
	~EventManager();
};
EventManager::EventManager()
{
}
void EventManager::recordComplaints()
{
}
void EventManager::manageStaff()
{
}
EventManager::~EventManager()
{
```

```
}
```

Report

```
class Report
{
private:
        int reportNo;
public:
        void generateEconomyReport(BanquetHall *b);
        void generateUsageReport();
        void generatePopularityReport();
        void displayReport();
};
```