Topic: Online Apartment Sale System

Group no : MLB_WE _01.02_05

Campus : Malabe

Submission Date : 20/05/2022

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**
IT21360428	Monali G.M.N.	0722415616
IT21388316	Rathnayake R.M.S.N.	0717180522
IT21362408	Munasinghe H.R.R	0712629373
IT21361722	Perera C.A.K.L	0774110053
IT21370700	Rathnathilaka R.G.I.N	0770340369

DESCRIPTION

With the development, people began to construct apartment complexes due to the shortage of land. After building them, owners needed an easy way to sell or rent their apartments. In order to solve this problem, an online apartment selling system can be introduced. Additionally, this system made it easier for home seekers.

This system is designed to make it facile for sellers to come and sell their own apartments. Also, through this system, buyers can facilely find an apartment suite to their budget.

An unregistered user (visitor) can visit the website and look for the available apartments. However, if the visitor wishes to get further information, such as the contact information of a seller, the visitor needs to register into the system using validating details. And a visitor who wants to post their apartment to sell on the website also needs to register to the system as a seller.

In this system, pictures and information related to each apartment are mentioned in its description. Therefore users might not be needed to meet the seller in person to check the apartment details and visit the apartment.

This system was designed with highly user-friendly functions so that users can quickly access information in the system.

REQUIREMENTS

1. First, a user has to register to the system.
2. User use details such as username and password to log in to the system.
3. Admins Check the validity of the user registration details.
4. On the home page, the System will display three options “Buy”, “Sell” and "Rent".
5. If the customer is a seller they can select the “Sell” option and go forward. There it is displayed to choose an apartment type as given in the system as “Select Type”. The system has categorized apartments according to their facilities. And then they are guided to enter their apartment details into the provided form in the system and can upload some detailed images of their apartment. Images should cover mainly the outer appearance, bedrooms, living room, dining room, and bathrooms. Following the completion of the details in the form, the system automatically generates an Apartment ID for your Apartment and then they can select the next option “Sell” where the system generates a report of the “Sell ID” and “Sell Date” on placement.

   If the seller intends not to sell the apartment at the moment they can save your details before going to the next step. If the user intends to sell now, the system will guide you to the payment portal where you should pay a small charge to place your apartment on the system. The payment portal checks the validity of payment details.

6. If the customer is a buyer, they can select the “Buy” Option and search for preferred apartment types and compare the Ratings of the previous customers who have bought them. Users can filter apartments using locations, prices, status, and types.
7. If the customer is a renter, they can select the “Rent” Option and go forward. You can follow the steps provided by the system and perform the required actions. The behavior of sellers and renters in the system is very similar.

NOUN / VERB ANALYSIS

Nouns

Verbs

1. First, a user has to register to the system.
2. User use details such as username and password to log in to the system.
3. Admin Check the validity of the user registration details.
4. On the home page, the System will display three options “Buy”, “Sell” and "Rent".
5. If the customer is a seller they can select the “Sell” option and go forward. There it is displayed to choose an apartment type as given in the system as “Select Type”. The system has categorized apartments according to their facilities. And then they are guided to enter their apartment details into the provided form in the system and can upload some detailed images of their apartment. Images should cover mainly the outer appearance, bedrooms, living room, dining room, and bathrooms. Following the completion of the details in the form, the system automatically generates an Apartment ID for your Apartment and then they can select the next option “Sell” where the system generates a report of the “Sell ID” and “Sell Date” on placement.

If the seller intends not to sell the apartment at the moment they can save your details before going to the next step. If the user intends to sell now, the system will guide you to the payment where you should pay a small charge to place your apartment on the system. The payment portal checks the validity of payment details.

6. If the customer is a buyer, they can select the “Buy” Option and search for preferred apartment types and compare the Ratings of the previous customers who have bought them. Users can filter apartments using locations, prices, status, and types.
7. If the customer is a renter, they can select the “Rent” Option and give details about their rental houses and go forward. You can follow the steps provided by the system and perform the required actions. The behavior of sellers and renters in the system is very similar.

IDENTIFIED CLASSES USING NOUN VERB ANALYSIS

User -	class
details -	Attributes
username -	Attributes
password -	Attributes
system -	the system itself (outside the scope)
Admin -	class
user registration details -	Attributes
home page -	the system itself (outside the scope)
customer -	Metalanguage
seller -	class
apartments -	class
apartment details -	Attributes
images -	Attributes
bedrooms -	Attributes
living room -	Attributes
dining room -	Attributes
bathrooms -	Attributes
Apartment ID -	Attributes
Sell ID -	Attributes
Sell Date -	Attributes
payment -	class

payment details - Attributes

buyer - class

apartment types - Attributes

locations - Attributes

prices - Attributes

status - Attributes

types - Attributes

renter - class

rental houses - class

Classes

1. User
2. Admin
3. seller
4. apartment
5. payment
6. buyer
7. renter
8. rental house

CRC CARDS

User	
Responsibility	Collaborators
Register to the System	
Register details	
Allow to view the apartment	Apartment
Allow to view the rental house	RentalHouse
Add and update user details	

Seller	
Responsibility	Collaborators
Add an apartment	Apartment
View buyer request	Buyer
Edit apartment details	Apartment
Pay advertise fees	Payment

Renter	
Responsibility	Collaborators
Add rental house	RentalHouse
View buyer request	Buyer
Edit rental house details	RentalHouse
Pay advertise fees	Payment

Apartment	
Responsibility	Collaborators
Apartment details	
Seller details	Seller

Payment	
Responsibility	Collaborators
checks the validity of the renter's payment details	Renter
checks the validity of the seller's payment details	Seller
payment details	

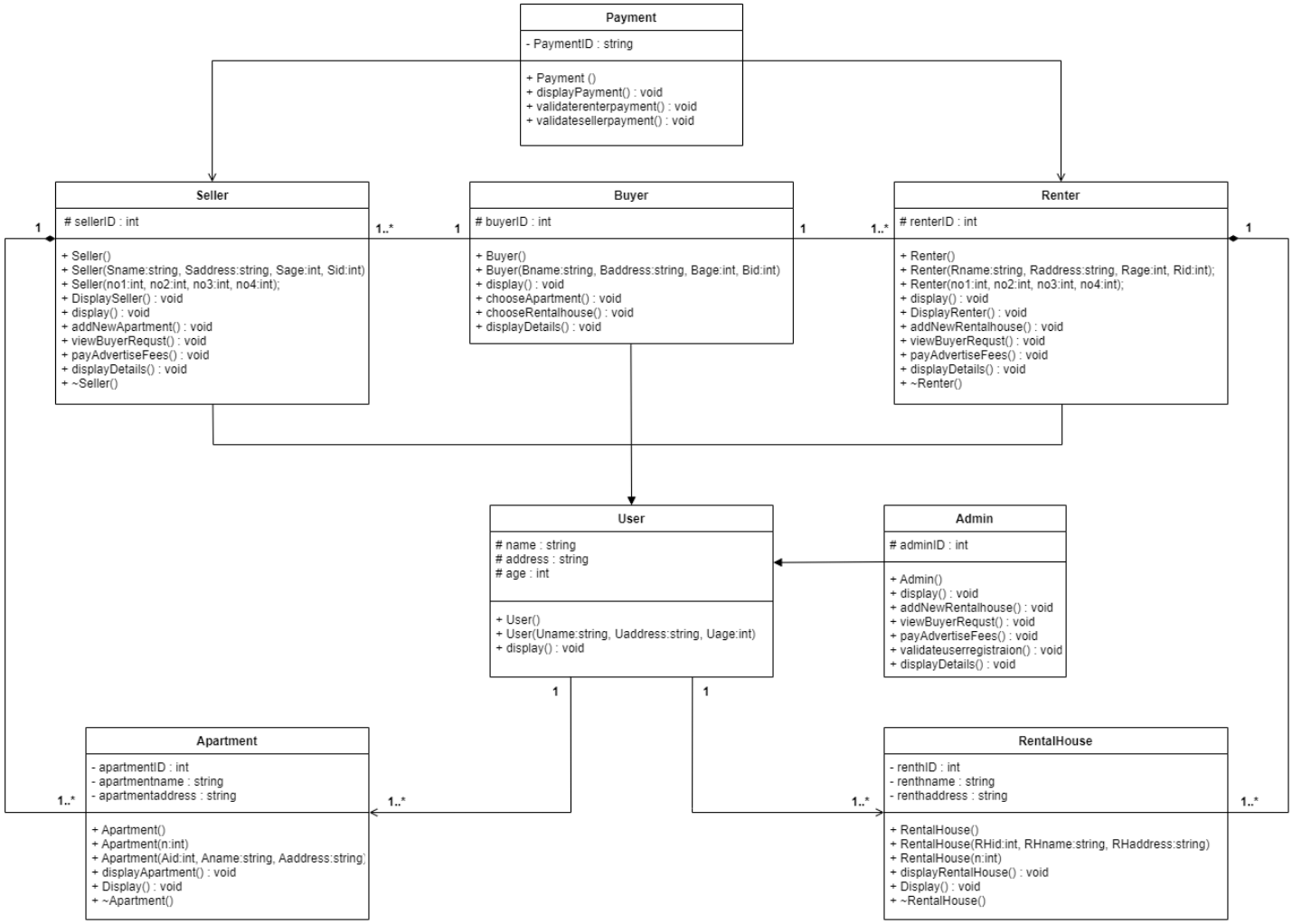
Buyer	
Responsibility	Collaborators
View apartment	
Choose an apartment	Apartment
Contact seller	Seller
Choose a rental house	RentalHouse
Contact renter	Renter

RentalHouse	
Responsibility	Collaborators
Rental house details	
Renter details	Renter

Admin	
Responsibility	Collaborators
Check the validity of the user registration details	user
Registration details	

Exercise 1

class diagram



Exercise 2

CODE IMPLIMENTATION

Definition of the class

```
#include <cstring>
#include <iostream>
#define SIZE 4
using namespace std;

class Apartment;
class RentalHouse;
class User;
class Seller;
class Buyer;
class Renter;
class Payment;
class Admin;

//Apartment class

class Apartment
{
private:
    int apartmentID;
    string apartmentname;
    string apartmentaddress;

public:
    Apartment();
    Apartment(int n);
```

```
    Apartment(int Aid, string Aname, string
Aaddress);
    void displayApartment();
    void Display();
    ~Apartment();
};

//RentalHouse class

class RentalHouse
{
private:
    int renthID;
    string renthname;
    string renthaddress;

public:
    RentalHouse();
    RentalHouse(int RHid, string RHname,
string RHaddress);
    RentalHouse(int n);
    void displayRentalHouse();
    void Display();
    ~RentalHouse();
};

//User class

class User {

protected:
    string name;
    string address;
```

```
    int age;

private:
    Apartment* app[SIZE]; // an object of
Apartmentr as attribute of User
    RentalHouse* rh[SIZE]; // an object of
RentalHouse as attribute of User

public:
    User() {}
    User(string Uname, string Uaddress, int
Uage);
    void display();
    void displayDetails();
};


//Seller class

class Seller : public User {
protected:
    int sellerID;

private:
    Buyer* buyer;  // an object of Buyer as
attribute of Seller
    Apartment* ap[SIZE]; // an object of
Apartment as attribute of Seller

public:
    Seller() {}
    Seller(string Sname, string Saddress,
int Sage, int Sid);
```

```
    Seller(int no1, int no2, int no3, int
no4);
    void DisplaySeller();
    void display();
    void addNewApartment();
    void viewBuyerRequst();
    void payAdvertiseFees();
    void displayDetails();
    ~Seller();
};

//Buyer class

class Buyer : public User {
protected:
    int buyerID;

private:
    Seller* seller[SIZE];  // an object of
Seller as attribute of Buyer
    Renter* r[SIZE];  // an object of
Renter as attribute of Buyer

public:
    Buyer() {}
    Buyer(string Bname, string Baddress,
int Bage, int Bid);
    void display();
    void chooseApartment();
    void chooseRentalhouse();
    void displayDetails();
};
```

```
//Renter class

class Renter : public User {
protected:
    int renterID;

private:
    Buyer* buy;  // an object of Buyer as
attribute of Renter
    RentalHouse* renh[SIZE];
public:
    Renter() {}
    Renter(string Rname, string Raddress,
int Rage, int Rid);
    Renter(int no1, int no2, int no3, int
no4);
    void display();
    void DisplayRenter();
    void addNewRentalhouse();
    void viewBuyerRequst();
    void payAdvertiseFees();
    void displayDetails();
    ~Renter();
};

//Payment class

class Payment
{
private:
    string PaymentID;
    Seller* sel; //an object of Seller as
attribute of Payment
```

```
    Renter* ren; //an object of Renter as
attribute of Payment

public:
    Payment() {}
    void displayPayment();
    void validaterenterpayment();
    void validatesellerpayment();
};

//Admin class

class Admin : public User {
protected:
    int adminID;

public:
    Admin() {}

    void display();
    void addNewRentalhouse();
    void viewBuyerRequst();
    void payAdvertiseFees();
    void validateuserregistraion();
    void displayDetails();
};
```

Implementation of the class

```
#include <cstring>
#include <iostream>
#include "classes.h"
#define SIZE 4

using namespace std;

//Apartment class
Apartment::Apartment()
{ }

Apartment::Apartment(int n)
{
    apartmentID = n;
}

Apartment::Apartment(int Aid, string
Aname, string Aaddress)
{
    apartmentID = Aid;
    apartmentname = Aname;
    apartmentaddress = Aaddress;
}

void Apartment::displayApartment()
{
    cout << " Apartment ID  = " <<
apartmentID << endl;
    cout << "Apartment name  = " <<
apartmentname << endl;
```

```
    cout << " Apartment Address  = " <<
apartmentaddress << endl;
}

void Apartment::Display() {
    cout << "Apartment id " << apartmentID
<< endl;
}

Apartment::~Apartment() {
    cout << "Deleting apartment  " <<
apartmentID << endl;
}

//RentalHouse class

RentalHouse::RentalHouse()
{ }

RentalHouse::RentalHouse(int RHid, string
RHname, string RHaddress)
{
    renthID = RHid;
    renthname = RHname;
    renthaddress = RHaddress;
}

RentalHouse::RentalHouse(int n)
{
    renthID = n;
}

void RentalHouse::displayRentalHouse()
```

```
{
    cout << " Rental house ID  = " <<
renthID << endl;
    cout << "Rental house name  = " <<
renthname << endl;
    cout << "Rental house location =" <<
renthaddress << endl;
}

void RentalHouse::Display() {
    cout << "Renthouse id " << renthID <<
endl;
}

RentalHouse::~RentalHouse() {
    cout << "Deleting Rental House  " <<
renthID << endl;
}

//User class
User::User()
{}
User::User(string Uname, string Uaddress,
int Uage) {
    name = Uname;
    address = Uaddress;
    age = Uage;
}

void User::display()
{
    cout << "this is User class" << endl;
}
```

```
void User::displayDetails()
{
    cout << name << " " << address << " "
<< age << endl;
}

//Seller class

Seller::Seller()
{ }

Seller::Seller(string Sname, string
Saddress, int Sage, int Sid) {
    name = Sname;
    address = Saddress;
    age = Sage;
    sellerID = Sid;
}

Seller::Seller(int no1, int no2, int no3,
int no4) {
    ap[0] = new Apartment(no1);
    ap[1] = new Apartment(no2);
    ap[2] = new Apartment(no3);
    ap[3] = new Apartment(no4);

}

void  Seller::DisplaySeller() {
    for (int i = 0; i < SIZE; i++)
        ap[i]->Display();
}
```

```
void Seller::display() {
    cout << "this is Seller class. "
        << "Derived class from User" <<
endl;
}

void Seller::addNewApartment()
{}

void Seller::viewBuyerRequst()
{ }

void Seller::payAdvertiseFees()

{}
void Seller::displayDetails() {
    cout << name << " " << address << " "
<< age << " " << sellerID << endl;
}

Seller::~Seller() {
    cout << "Seller object closing " <<
endl;
    for (int i = 0; i < SIZE; i++)
        delete ap[i];
    cout << "Everthing is deleted in
Apartment class" << endl;
}

//Buyer class

Buyer::Buyer()
```

```
{}

Buyer::Buyer(string Bname, string
Baddress, int Bage, int Bid) {
    name = Bname;
    address = Baddress;
    age = Bage;
    buyerID = Bid;
}

void Buyer::Buyer::display() {
    cout << "this is the Buyer class. "
        << "Derived class from User. " <<
endl;
}

void Buyer::chooseApartment()
{}

void Buyer::chooseRentalhouse()
{}

void Buyer::displayDetails() {
    cout << name << " " << address << " "
<< age << " " << buyerID << endl;
}

//Renter class

Renter::Renter()
{}
```

```
Renter::Renter(string Rname, string
Raddress, int Rage, int Rid) {
    name = Rname;
    address = Raddress;
    age = Rage;
    renterID = Rid;
}

Renter::Renter(int no1, int no2, int no3,
int no4) {
    renh[0] = new RentalHouse(no1);
    renh[1] = new RentalHouse(no2);
    renh[2] = new RentalHouse(no3);
    renh[3] = new RentalHouse(no4);

}

void  Renter::display() {
    cout << "this is the Renter class. "
        << "Derived class from User. " <<
endl;
}

void Renter::DisplayRenter() {
    for (int i = 0; i < SIZE; i++)
        renh[i]->Display();
}
void Renter::addNewRentalhouse()
{}

void Renter::viewBuyerRequst()
{}
```

```
void Renter::payAdvertiseFees()
{}

void Renter::displayDetails() {
    cout << name << " " << address << " "
<< age << " " << renterID << endl;
}

Renter::~Renter() {
    cout << "Renter object closing " <<
endl;
    for (int r = 0; r < SIZE; r++)
        delete renh[r];
    cout << "Everthing is deleted in
Rental house class" << endl;
}

//Payment class

Payment::Payment()
{}

void Payment::displayPayment()
{
    cout << "This is a payment class " <<
endl;
}

void Payment::validaterenterpayment()
{}

void Payment::validatesellerpayment()
{}
```

```
//Admin class

Admin::Admin()
{}

void Admin::display() {
    cout << "this is the Admin class. "
        << "Derived class from User. " <<
endl;
}

void Admin::addNewRentalhouse()
{}

void Admin::viewBuyerRequst()
{}

void Admin::payAdvertiseFees()
{}

void Admin::validateuserregistraion()
{}

void Admin::displayDetails() {
    cout << name << " " << address << " "
<< age << " " << adminID << endl;
}
```

Main program

```
#include <cstring>
#include <iostream>
#include "classes.h"
#define SIZE 4
using namespace std;


int main() {

    User u1("Kamal", "Kurunegala", 22);
    u1.display();
    u1.displayDetails();

    cout << endl << endl << "
**************************" << endl <<
endl;

    Apartment* a = new Apartment(12,
"Araliya", "Colombo 7");
    a->displayApartment();

    cout << endl << endl << "
**************************" << endl <<
endl;

    RentalHouse* r = new RentalHouse(34,
"Rent land", "Kaduwela");
    r->displayRentalHouse();
```

```
    cout << endl << endl << "
**************************" << endl <<
endl;

    Payment p1;
    p1.displayPayment();

    cout << endl << endl << "
**************************" << endl <<
endl;

    Seller* se;
    se = new Seller(305, 506, 45, 43);
    se->DisplaySeller();
    delete se;

    cout << endl << endl << "
**************************" << endl <<
endl;

    Renter* re;
    re = new Renter(45, 34, 67, 37);
    re->DisplayRenter();
    delete re;

    return 0;
}
```