



Topic : Wedding planning system

Group no: KDY_05

Campus : Kandy

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
IT21265242	Zimmendra C.H	0769654581
IT21294952	Aluthge I. N	0704208122
IT21200410	Kiriharan M	0779503160
IT21213212	Weerasekara D.G.D.I.	0755620819
it21276446	Wegodapola A. R	0779461808

Exercise 1**1)**

- A user needs to register to the wedding planning system by providing details such as bride's name, groom's name, No of guests, Wedding date, Email address and a contact no. These details will be stored on the edit or delete database with a unique ID.
- A registered customer that has forgotten their password can use forget password function to reset and add a new password.
- Admin will be able to manage vendors, customers, categories, and customer feedbacks
- The vendor will be able to request the admin to publish their advertisement
- Once the request is accepted, the vendor is given a unique Vendor ID
- Customers can leave feedbacks that can be a complaint or a review with the description
- Customer selects a category
- The customer selects a package for the specific his/her specific needs and selects the package of the given.
- The customer can view the added products to the cart showing relevant prices, quantity, and the total price
- The customer then will be able to add products of their choice to the cart
- Customer can add products in their cart according to their choice
- Customer can update existing added items
- Customer can then proceed to checkout to add payment details including payment method and card details, system will validate this payment
- Once the order is placed the specific order is given a unique Order ID
- User and admin can change their settings
- When a customer clicks on a category, the category records number of clicks in the database and the admin can access the data for further marketing purposes.

2)

Classes

- User
- Category
- Feedback
- Cart
- Order
- Admin
- Customer
- Package
- Payment

3)

Customer	
Responsibilities	Collaboration
Register details	
leave feedbacks	Feedbacks
view feedbacks	Feedbacks
delete feedback	Feedbacks
select packages	Packages
select categories	Category

Category	
Responsibilities	Collaboration
Display details	
Record number of clicks	

Feedback	
Responsibilities	Collaboration
Store feedback Display feedback	

Package	
Responsibilities	Collaboration
Store package details Display package	

Cart	
Responsibilities	Collaboration
Add items Delete items Update items Calculate price	

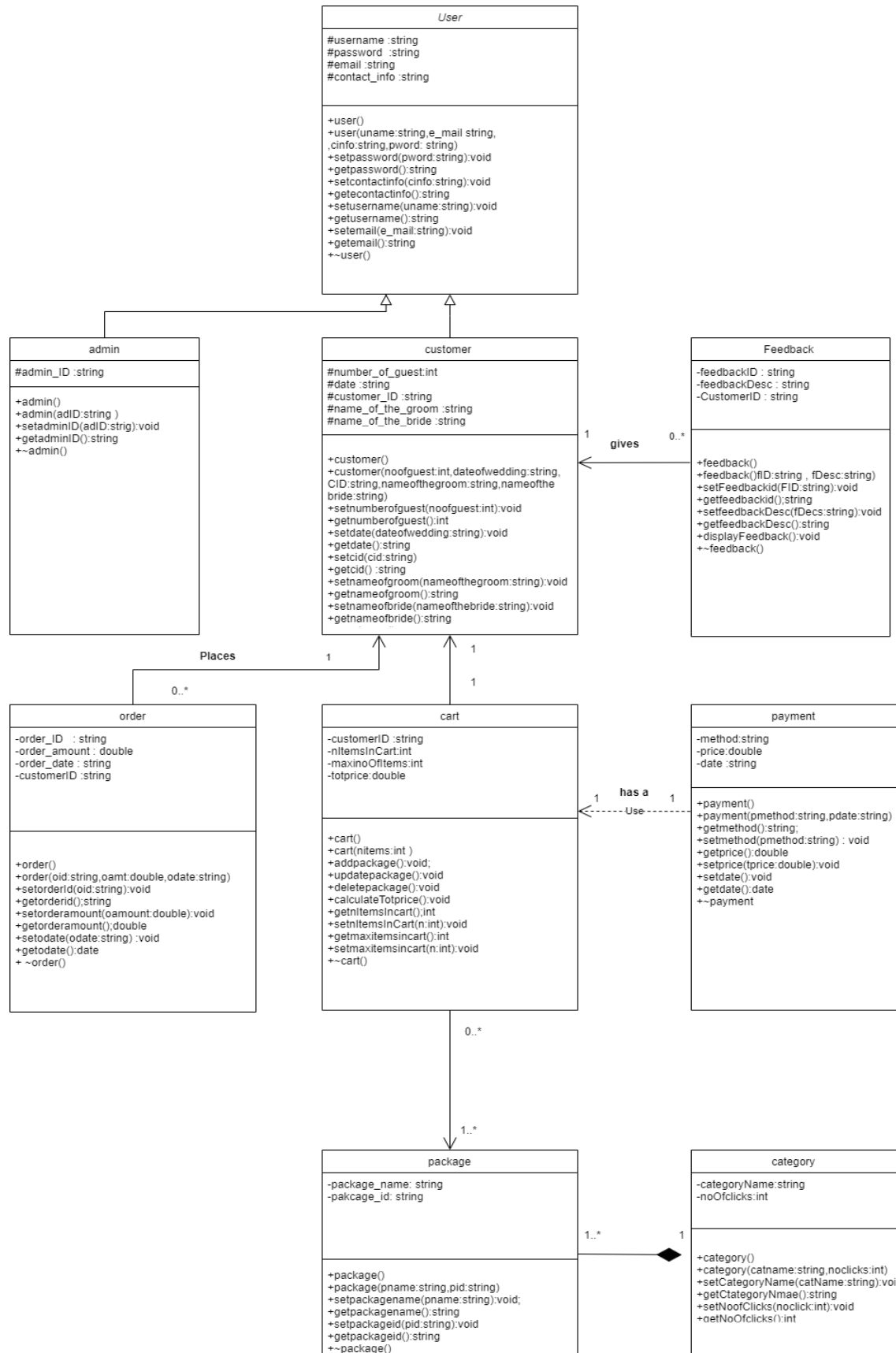
User	
Responsibilities	Collaboration
Store user details	

Order	
Responsibilities	Collaboration
Store order	
Add order	
Display order details	

Admin	
Responsibilities	Collaboration
Store admin details	

Payment	
Responsibilities	Collaboration
Validate payment Display total amount Display payment methods	Cart

4)



Package – Zimmendra C.H

Order - Aluthge I.N it21294952

Cart , Payment - Kiriharan M

User, admin, Customer - Weerasekara D.G.D.I. it21213212

Category, Feedback - Wegodapola A.R it21276446

Exercise 2

IT21213212 Weerasekara D.G.D.I

User

```
user.h ×
1  #pragma once
2
3  //create class
4  class User
5  {
6  protected:
7      char username[15];
8      char password[8];
9      char email[40];
10     char contact_info[10];
11
12 public:
13     User(); // constructor
14     User(char uname[], char pword[], char e_mail[], char cinfo[]); // overloaded constructor for registration
15
16     void setpassword(char pword[]); // setter for password
17     char* getpassword(); // getter for password
18
19     void setcontactinfo(char cinfo[]); // setter for contact info
20     char* getcontactinfo(); // getter for contact info
21
22     void setusername(char uname[]); // setter user name
23     char* getusername(); // getter for user name
24
25     void setemail(char e_mail[]); // setter for e-mail
26     char* getemail(); // getter for e-mail
27
28     ~User(); // destructor called for registered user
29
30 };
```

Admin

```
admin.h x
1 #include "user.h"
2
3 //create class
4 class Admin : public User
5 {
6     protected:
7         char admin_ID[6];
8
9     public:
10        Admin(); // constructor called for registered Admin
11        Admin(char adID[], char username[], char password[], char email[], char contact_info[]); // overloaded constructor for
            registration
12
13        void setadminID(char adID[]); // setter for admin ID
14        char* getadminID(); // getter for admin ID
15
16        ~Admin(); // destructor called for registered Admin
17 };
```

Customer

```
customer.h x
1 #pragma once
2 #include "user.h"
3 //include "feedback.h"
4 //include "order.h"
5 //include "cart.h"
6
7 //create the class
8 class Customer : public User
9 {
10     protected:
11         int number_of_guest;
12         char customer_ID[5];
13         char date[10];
14         char name_of_the_groom[35];
15         char name_of_the_bride[35];
16
17         //Cart* Cart;
18         //Feedback* Feedback;
19
20     public:
21         Customer(); // constructor
22         Customer(int noofguest, char CID[], char dateofwedding[], char nameofthegroom[], char nameofthebride[], char username[], char password[], char email[], char
            contact_info[]); // overloaded constructor for registration
23
24         void setnumberofguest(int noofguest); // setter for number of guest
25         int getnumberofguest(); // getter for number of guest
26
27         void setdate(char dateofwedding[]); // setter for date
28         char* getdate(); // getter for date
29
30         void setnameofgroom(char nameofthegroom[]); // setter for name of the groom
31         char* getnameofgroom(); // getter for name of the groom
32
33         void setnameofbride(char nameofthebride[]); // setter for name of the bride
34         char* getnameofbride(); // getter for name of the bride
35
36         void setcid(char cid[]); // setter for customer ID
37         char* getcid(); // getter for customer ID
38
39         ~Customer(); // destructor called for registered customer
40 };
```

IT212994952 Aluthge I.N

Order

```
Order.h x
1 #include "customer.h"
2 class Order
3 {
4 private:
5     char Order_ID[5];
6     double Order_Amount;
7     char Order_Date[8];
8     char Customer_ID[5];
9     Customer *Cust; //the Customer ID assigned to each order placed by a particular customer
10
11 public:
12     Order(); //default constructor
13     Order(char OID[], double Oamt, char ODate[], Customer *Cust); //overloaded constructor
14     void setOrderID(char[]); //setter for Order ID
15     char getOrderID(); //getter for Order ID
16     void setOrder_Amount(double Oamt); //setter for Order Amount
17     double getOrder_Amount(); //getter for Order Amount
18     void setODate(char[]); //setter for Order Date
19     char getODate(); //getter for Order Date
20     ~Order(); //Destructor
21 };
```

IT21200410 Kiriharan M

Cart

```

Cart.h x
1  #include "customer.h"
2  #include "package.h"
3  #define SIZE 6
4
5  // Implementation of the Class Cart
6  // Cart class has a one way association relationship with customer
7  // Cart class has a one way association relationship with the packages class
8  // Cart class has a dependency relationship with the payment class(payment class is dependent on the cart class)
9
10 class Cart
11 {
12     private:
13         char customerID[6];    // stores customer ID
14         int nItemsInCart;      // Stores number of items currently in cart
15         int maxItemsInCart;    // Stores max number of items that can be stored in a class
16         double totPrice;       // Stores total price of packages added in class
17         Customer *c1;          // Implementing one way association
18         package *cartItems[SIZE]; // Array to store added packages
19
20     public:
21         Cart();                // Default Constructor
22         Cart(Customer *pc1, int nItems); // Overloaded Constructor
23         void addPackage(package *p1); // function to add Packages to cart
24         void updatePackage(package *p1); // function to update Packages in cart
25         void deletePackage(package *p1); // function to delete Packages in cart
26         void calculateTotPrice(); // function to calculate totalPrice of Packages
27
28         int getnItemsInCart(); // Accessor to get number of items in cart
29         void setnItemsInCart(int n); // Mutator to set number of items in cart
30         int getmaxItemsInCart(); // Accessor to get max of Items in cart
31         void setmaxItemsInCart(int n); // Mutator to set number of items in cart
32         double getTotPrice(); // Accessor to get total Price of all Packages
33         ~Cart();             // Cart destructor
34
35 };

```

Payment

```

Payment.h x
1  #include "Cart.h"
2  // Implementation of class Payment
3  // Payment class has a dependency relationship with the cart class(Payment class is dependant on cart class)
4  class Payment
5  {
6     private:
7         char method[20];      // Stores chosen payment method
8         double price;         // Stores price to pay
9         char date[10];        // Stores payment date
10
11     public:
12         Payment();            // Default constructor of Payment class
13         Payment(char pMethod[], Cart* c, char pdate[]); // Overloaded constructor of payment class
14
15         char* getMethod();    // Accessor to get Payment method
16         void setMethod(char* pmethod[]); // Mutator to set Payment method
17         double getPrice();    // Accessor to get payment price
18         void setPrice(double pPrice); // Mutator to set total Price
19         char* getDate();      // Accessor to get payment Date
20         void setDate(char* pDate); // Mutator to set payment price
21
22         ~Payment();           // Payment destructor
23
24 };

```

IT21265242 Zimmendra .C.H

Package

```
package.h ×
1 //create class
2 class package
3 {
4 private:
5
6     char package_name[10];
7     char package_id[10];
8
9 public:
10    package();//constructor
11    package(char pname[], char pid[]);//overloaded constructor
12    void setpackagename(char pname[]);//setter
13    char getpackagename();//getter
14
15    void setpackageid(char pid[]);//setter
16    char getpackageid();//getter
17
18    ~package();//destructor
19 };
20
21
```

IT21276446 Wegodapala A.R

Feedback

```
feedback.h ×
1 #include "customer.h"
2
3 class feedback{
4 private:
5     char customerID[5];
6     char feedbackID[5];
7     char feedbackDesc[500];
8     Customer *cus;
9 public:
10    feedback();//Default Constructor
11    feedback(char fID[], char fDesc[], Customer *pCus); //Overloaded constructor
12    void setFeedbackId(char fID[]);
13    char getFeedbackId();
14    void setFeedbackDesc(char fDesc[]);
15    char getFeedbackDesc();
16    void displayFeedback();
17    ~feedback(); //Destructor
18 };
```

Category

```
category.h ×
1 #include "package.h"
2 #define SIZE 1
3 class category{
4
5     private:
6         package *packages[SIZE];
7         char categoryName[20];
8         int noOfClicks;
9
10    public:
11        category();
12        category(char catName[], int noClicks, char pkid[], char pkname1[]);
13        void setCategoryName(char catName[]);
14        char getCategoryName();
15        void setNoofClicks(int noClicks);
16        int getNoofClicks();
17        void displayCategory();
18        ~category();
19
20 };
```

Main.cpp

```
main.cpp ×
1 #include "Payment.h"
2 #include "category.h"
3 #include "admin.h"
4 #include "feedback.h"
5 #include "Order.h"
6 #include <iostream>
7 using namespace std;
8
9 int main()
10 {
11     // Declaration of an dynamic object using class Admin
12     Admin *admin1 = new Admin((char*) "A0001", (char*) "Zimmendra", (char*) "Zima1234!", (char*) "zima@gmail.com", (char*) "0769654581");
13
14     // Declaration of an dynamic object using class Customer
15     Customer *cust1 = new Customer(150, (char*) "ID001", (char*) "2022-05-19", (char*) "Udayanga", (char*) "Imandi", (char*) "ImandiUdayanga", (char*) "Imandi123", (char*)
16     "imandiudayanga@gmail.com", (char*) "0704208122");
17
18     // Declaration of an dynamic object using class Feedback
19     feedback *feed1 = new feedback((char*) "FD001", (char*) "Loved it", cust1);
20
21     // Declaration of an dynamic object using class Order
22     Order *order1 = new Order((char*) "O0001", 100000, (char*) "2022-05-19", cust1);
23
24     // Declaration of an dynamic object using class Cart
25     Cart *c1 = new Cart(cust1, 0);
26
27     // Declaration of an dynamic object using class Payment
28     Payment *p1 = new Payment((char*) "Paypal", c1, (char*) "2022-05-18");
29
30     // Declaration of an dynamic object using class Package
31     package *pack1 = new package((char*) "Hotel and Food", (char*) "PD001");
32
33     // Declaration of an dynamic object using class Package
34     category *category1 = new category((char*) "Florist", 20, (char*) "PK001", (char*) "Package 1");
35
36     delete admin1;
37     delete cust1;
38     delete feed1;
39     delete order1;
40     delete c1;
41     delete p1;
42     delete pack1;
43     delete category1;
44 }
```